EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

HAWK-I Pipeline User Manual

VLT-MAN-ESO-19500-4407

Issue 2.5.10

Date February 14, 2025

	Name	Date	Signature
Prepared:	UK In-kind Team	February 14, 2025	
Approved:	P.Ballester		
Released:	M. Peron		

This page was intentionally left blank

Change record

Issue/Rev.	Date	Section/Parag. affected	Reason/Initiation/Documents/Remarks
2.2.0	01/04/2017	All	Pipeline Update
2.4.2	06/01/2018	See §3	NEWS section added
2.4.3	01/06/2018	See §3	
2.4.6	01/05/2019	See §3	
2.4.7	01/04/2020	See §3	
2.4.8	03/06/2020	See §3	
2.4.9	28/01/2021	See §3	
2.4.10	28/01/2021	See §3	
2.4.11	04/05/2022	See §3	
2.4.12	14/03/2023	See §3	
2.4.13	10/05/2023	See §3	
2.4.14	14/12/2023	See §3	
2.4.15	08/07/2024	See §3	
2.5.7	13/07/2024	See §3	
2.5.8	09/08/2024	See §3	
2.5.9	23/09/2024	See §3	
2.5.10	14/02/2025	See §3	

This page was intentionally left blank

Contents

1 Introduction **9**

1.1 Purpose . . . 9

1.2 Acknowledgements . . . 9

1.3 Scope . . . 9

1.4 Conventions on Style and Notation . . . 9

1.5 Reference documents . . . 10

2 Overview **12**

2.1 The HAWK-I Instrument . . . 12

2.2 The HAWK-I Data Reduction Pipeline . . . 12

3 What's new **14**

3.1 What's new in the latest pipeline release . . . 14

3.2 v2.5.10 . . . 14

3.3 v2.5.9 . . . 14

3.4 v2.5.8 . . . 14

3.5 v2.5.7 . . . 14

3.6 v2.4.15 . . . 14

3.7 v2.4.14 . . . 14

3.8 v2.4.13 . . . 14

3.9 v2.4.12 . . . 14

3.10 v2.4.11 . . . 15

3.11 v2.4.10 . . . 15

3.12 v2.4.9 . . . 15

3.13 2.4.8 . . . 15

3.14 2.4.7 . . . 15

3.15 2.4.6 . . . 15

3.16 What's new in pipeline release 2.4.3 . . . 15

3.17 What's new in pipeline release 2.4.2 . . . 16

4 Installation **17**

4.1 System Requirements . . . 17

4.2 Installing the Software . . . 17

4.3 EsoRex, Reflex, and Gasgano . . . 18

5 Data Description **19**

5.1 Raw Data . . . 19

5.2 Static Calibration Data . . . 19

5.3 Photometric and Astrometric Catalogues . . . 21

5.4 Pipeline Products . . . 22

5.4.1 Naming convention for files . . . 22

5.4.2 Images . . . 22

5.4.3 Catalogues . . . 23

5.4.4 Diagnostic tables . . . 23

6 Data Reduction **24**

6.1 Getting Started with EsoRex . . . 24

6.2 Data Organization, Classification, & Association . . . 25

6.2.1 Header Keywords . . . 25

6.2.2 Classification . . . 26

6.2.3 Organization . . . 29

6.2.4 Association . . . 29

6.3 Data Reduction Cascade . . . 31

6.4 Calibration . . . 31

6.4.1 Dark . . . 31

6.4.2 Flat Field . . . 35

6.4.3 Detector Noise [optional] . . . 40

6.5 Standard Star Observations . . . 44

6.6 Science Observations . . . 47

6.6.1 Creating a stacked image with hawki_science_process . . . 48

6.6.2 Creating a tiled image with hawki_science_postprocess . . . 52

7 Troubleshooting **57**

7.1 Log file . . . 57

7.2 Sky subtraction . . . 57

7.3 Astrometry in crowded fields . . . 57

8 Recipe Reference **58**

8.1 Common recipe QC parameters . . . 58

8.2 hawki_dark_combine . . . 58

8.2.1 Description . . . 58

8.2.2 Input frames . . . 58

8.2.3 Recipe parameters . . . 58

8.2.4 Product frames . . . 60

8.2.5 Quality control parameters . . . 61

8.3 hawki_twilight_flat_combine . . . 62

8.3.1 Description . . . 62

8.3.2 Input frames . . . 62

8.3.3 Recipe parameters . . . 63

8.3.4 Product frames . . . 65

8.3.5 Quality control parameters . . . 66

8.4 hawki_detector_noise . . . 67

8.4.1 Description . . . 67

8.4.2 Input frames . . . 67

8.4.3 Recipe parameters . . . 67

8.4.4 Product frames . . . 68

8.4.5 Quality control parameters . . . 68

8.5 hawki_standard_process . . . 68

8.5.1 Description . . . 68

8.5.2 Input frames . . . 68

8.5.3 Recipe parameters . . . 69

8.5.4 Product frames . . . 71

8.5.5 Quality control parameters . . . 72

8.6 hawki_science_process . . . 75
8.6.1 Description . . . 75
8.6.2 Input frames . . . 75
8.6.3 Recipe parameters . . . 76
8.6.4 Product frames . . . 80
8.6.5 Quality control parameters . . . 82
8.7 hawki_science_postprocess . . . 84
8.7.1 Description . . . 84
8.7.2 Input frames . . . 85
8.7.3 Recipe parameters . . . 86
8.7.4 Product frames . . . 88
8.7.5 Quality control parameters . . . 89

9 Recipe Algorithms 92
9.1 Stacking . . . 92
9.2 Variance maps . . . 92
9.3 Confidence maps . . . 93
9.4 WCS fitting . . . 93
9.5 Detector readnoise and gain . . . 93
9.6 Object detection/catalogue generation . . . 94
9.6.1 Background analysis and object detection . . . 94
9.6.2 Object parameterisation . . . 94
9.6.3 Morphological classification . . . 96
9.7 Sky Estimation . . . 97
9.8 Nebulising . . . 98

A File Formats 99
A.1 Static Calibration Files . . . 99
A.2 Astrometric and Photometric Catalogue Files . . . 101
A.3 Recipe Product Catalogues . . . 102

1 Introduction

1.1 Purpose

The HAWK-I pipeline is a subsystem of the *VLT Data Flow System* (DFS). It is used in two operational environments, for the ESO *Data Flow Operations* (DFO), and for the *Paranal Science Operations* (PSO). It is used to provide a quick-look assessment of data, to generate master calibration data, to reduce of scientific exposures, and to assessed data quality. Additionally, the HAWK-I pipeline recipes are made public to the user community, to allow a more personalised processing of the data from the instrument. The purpose of this document is to describe a typical HAWK-I data reduction sequence with the HAWK-I pipeline.

This manual is a complete description of the data reduction recipes implemented by the the HAWKI pipeline, reflecting the status of the HAWKI pipeline version 2.5.10.

1.2 Acknowledgements

The HAWK-I pipeline is based on the VISTA Data Reduction Software developed by the Cambridge Astronomy Survey Unit at the University of Cambridge.

This release has benefitted from the feedback provided by ESO staff. In particular we would like to thank M. Neeser, L. Coccato, and A. Gabasch for testing the pipeline and their suggestions for improvements.

1.3 Scope

The examples on running individual pipeline recipes in this manual use the `esorex` command and manually created list of input files. Several interfaces to automatically organise the data, create the list of input files and execute the pipeline recipes in the proper sequence are available, see the *https://www.eso.org/pipelines* for details.

Please note that the use of *Gasgano* as a GUI for processing data is deprecated. Its use is no longer recommended and the related section in this manual, as well as support for *Gasgano* as a data processing GUI application in general will be dropped entirely in a future release.

1.4 Conventions on Style and Notation

This document uses the following conventions for font styles:

- **bold**: commands or other input to be typed as shown
- *italics*: placeholder of input to be replaced by real text as appropriate for particular use
- `teletype`: FITS keywords, program names, file paths, and terminal output, etc.

A generic shell prompt is used as a prefix to commands or other user input.

To improve readability, hierarchical FITS keyword names appear in 'dot-notation'. This means that the prefix `"HIERARCH ESO"` is omitted and the spaces separating the keyword name in the FITS header are replaced by a single dot, e.g. `DET.DIT` refers to the keyword `"HIERARCH ESO DET DIT"`.

1.5 Reference documents

[1] R. M. Cutri and et al. VizieR Online Data Catalog: AllWISE Data Release (Cutri+ 2013). *VizieR Online Data Catalog*, 2328:0, January 2014. 21

[2] R. M. Cutri, M. F. Skrutskie, S. van Dyk, C. A. Beichman, J. M. Carpenter, T. Chester, L. Cambresy, T. Evans, J. Fowler, J. Gizis, E. Howard, J. Huchra, T. Jarrett, E. L. Kopan, J. D. Kirkpatrick, R. M. Light, K. A. Marsh, H. McCallon, S. Schneider, R. Stiening, M. Sykes, M. Weinberg, W. A. Wheaton, S. Wheelock, and N. Zacarias. VizieR Online Data Catalog: 2MASS All-Sky Catalog of Point Sources (Cutri+ 2003). *VizieR Online Data Catalog*, 2246:0, June 2003. 21

[3] ESO. *HAWK-I Reflex Tutorial*, 2.5.10edition, 2015. VLT-MAN-ESO-XXXXX-XXXX. 12, 18

[4] ESO/SDD/DFS, *http://www.eso.org/cpl/esorex.html*. *ESOREX home page*. 24

[5] ESO/SDD/DFS, *http://www.eso.org/gasgano/*. *Gasgano User's Manual*. VLT-PRO-ESO-19000-1932. 12

[6] M. J. Irwin. Automatic analysis of crowded fields. *MNRAS*, 214:575–604, June 1985. 94

[7] D. J. Schlegel, D. P. Finkbeiner, and M. Davis. Maps of Dust Infrared Emission for Use in Estimation of Reddening and Cosmic Microwave Background Radiation Foregrounds. *Astrophysical Journal*, 500:525–553, June 1998. 21, 99

[8] Forchì V. *Reflex User's Manual*. ESO/SDD/DFS, 0.7 edition, 2012. VLT-MAN-ESO-19000-5037. 12, 18

2 Overview

2.1 The HAWK-I Instrument

The High Acuity Wide-field K-Band Imager (HAWK-I) is a cryogenic wide-field infrared imager installed at the Nasmyth A focus of UT4. The on-sky field of view is 7.5′ by 7.5′, with a cross-shaped gap of 15″ between the four HAWAII 2RG 2048x2048 pixel detectors. The pixel scale is 0.106 ″ pix^{-1}. The instrument is offered with 10 observing filters placed in two filter wheels: 4 broad band filters (Y, J, H and K_s) and 6 narrow band filters (Br γ, CH_4, H_2, 1.061 μm, 1.187 μm and 2.090 μm).

Please refer to the instrument web page at http://www.eso.org/sci/facilities/paranal/instruments/hawki.html for detailed information about the instrument.

2.2 The HAWK-I Data Reduction Pipeline

The HAWK-I data reduction pipeline is part of a suite of tools provided by the Pipeline Systems Group (PPS) of the Science Operation Software Department at ESO. In collaboration with instrument consortia, the PSD implements data reduction pipelines for three main purposes:

Master calibration product creation: pipelines are used to produce master calibration products (e.g., combined bias frames, master flats).

Data quality control: pipelines are used to produce the quantitative information necessary to monitor instrument performance.

Science product creation: using pipeline-generated master calibration products, science products are produced for the supported instrument modes (e.g., calibrated images; stack of jittered images). The HAWK-I pipeline aims to provide scientific grade data products.

Instrument pipelines consist of a set of data processing modules that can be executed in one of four ways: 1) from the automatic data management tools available on Paranal, 2) from the command line (using *EsoRex*), 3) from *Reflex*, or 4) from *Gasgano*. The two latter applications are included in the pipeline distribution (see [8] and [5]). These applications can also be downloaded separately from http://www.eso.org/cpl/esorex.html, http://www.eso.org/reflex, and http://www.eso.org/gasgano. An illustrated guide to processing HAWK-I data with Reflex is provided in [3].

Note that this pipeline can only be used to process HAWK-I data in 'normal' imaging mode. It cannot process data taken in "Burst" or "Fast Jitter" mode.

Also note that the pipeline **is not designed to be used to process standard star fields or science images that span more than one Observation Block**. Attempting to process data that span multiple OBs may yield unexpected and/or less than ideal results, particularly with respect to sky corrections.

The HAWK-I pipeline comprises 7 recipes: 3 recipes for calibration, 1 recipe for processing images of photometric standard star fields, and 2 recipes for processing science images.

The three calibration recipes are:

hawki_dark_combine: create a master dark image by combining raw dark images. The recipe can also quantitatively compare the master dark to a user-specified 'reference' dark.

hawki_twilight_flat_combine: create a master flat image, a bad pixel map, and confidence map by combining raw twilight images. The recipe can also quantitatively compare the master dark to a user-specified 'reference' flat.

hawki_detector_noise: calculate the detector readout noise and gain from a set of 2 raw dark images and 2 raw twilight flat images.

The standard star calibration recipe is:

hawki_standard_process: create a calibrated images and object catalogues from a set of (non-jittered) standard star images.

The two science recipes are:

hawki_science_process: create calibrated images and object catalogues by combining a number of jittered science images on a chip-by-chip basis, i.e. a "stack".

hawki_science_postprocess: create one calibrated image and object catalogue from a set of jittered science images by combining images from all chips, i.e. a "tile". If requested, the recipe will "nebulise" the image when creating the object catalogue, i.e. remove the effect of strongly varying diffuse background emission.

More details on the pipeline and how to use it can be found later in this manual in §6.

3 What's new

3.1 What's new in the latest pipeline release

3.2 v2.5.10

- Public release EDPS workflows
- Improve description of local astrom/photom catalogues.

3.3 v2.5.9

- Change of Period October 2024 Release
- Remove calchecker metatarget from optional calibration (DFS-21389).
- Harmonize non-linearity reports between instruments (PIPE-11645).

3.4 v2.5.8

- Fix esoreflex bug for matplotlib 3.7.0+, PIPE-11679.

3.5 v2.5.7

- Add EDPS suport and ADARI reports, PIPE-11689.

3.6 v2.4.15

- Fix esoreflex bug for matplotlib 3.7.0+, PIPE-11679.

3.7 v2.4.14

- Release version for PIPE-10549 (QC parameters update)

3.8 v2.4.13

- Public Release 2023

3.9 v2.4.12

- PSO/QC Release 2023

3.10 v2.4.11

- RecipeLooper parameter added
- Manual formatting updates
- Corrections to zeropoint header keywords (CASU)

3.11 v2.4.10

- Updates the User and Pipeline Manuals
- Remove deprecated header keywords from products

3.12 v2.4.9

- Latest version of `m4macros` have been incorporated.

3.13 2.4.8

- Color terms added to Pipeline Manual.

3.14 2.4.7

- Reflex python scripts have been ported to python 3 keeping the backward compatibility with python 2.

3.15 2.4.6

- The pipeline has been built on many different platforms with different compilers and the warnings reported by those compiler were fixed. In a few occasions, the calculation of the absolute value of a number was done by using the wrong c-function (`fabs()` vs. `abs()`). This has been fixed. As a consequence, some values in the catalogues will show subtle differences.
- QC parameters that where not DICD compliant were fixed, i.e. QC.RON.XXX and QC.GAIN.XXX

3.16 What's new in pipeline release 2.4.3

- The reflex workflow has been updated to work with adaptive optics data. A problem in two interactive plotting routines with the presence of additional extensions for adaptive optics data has been fixed.
- The reflex workflow has been updated to make sure that parameter values specified by user are used in all future executions of a recipe if requested by user (parameters in interactive actors were updated too late in previous esoreflex versions).

- The reflex workflow has been updated to use the same time-stamps for the execution names in the Product Explorer and the file names on disk.
- All interactive recipe executions have been modified so that values of parameters are saved and reused in subsequent reduction of the same data set (setting of parameters from data base). Please note that this feature has been deactivated in the current workflow as it is not working with esoreflex 2.9. After esoreflex has been updated it can be activated in the workflow.
- An option to trigger the product explorer without running the data reduction cascade has been added (default: deactivated).
- The header keyword CASUEXPT has been renamed into EFF_EXPT
- The calculation of the IDP keywords ABMAGLIM and ABMAGSAT have been revised and changed.
- The header keyword QC.LIMITING_MAG is set to the keyword ABMAGLIM

3.17 What's new in pipeline release 2.4.2

- The pipeline has been updated to work with cpl version 7.1
- The reflex workflow has been updated to work with the newest esoreflex engine 2.9.
- The recipes have been updated to be able to process adaptive optic data with more than 4 extensions.
- Some compiler warnings have been removed and memory leaks were closed.
- A problem related to observations with dedicated sky has been solved: If dedicated sky observations are used for the data reduction certain recipe parameters caused an error leading to a recipe failure.

4 Installation

ESO pipelines can be installed via several methods, depending on your OS, most of which facilitate easy installation, upgrade and removal. Please see the "ESO Data Reduction Pipelines and Workflow Systems" page (*https://www.eso.org/pipelines*).

4.1 System Requirements

Due to the large number of files with short exposures, data reduction for infrared imaging may use a significant amount of memory. This can be exacerbated when the images are combined into a tile that covers a large part of the sky. The amount of memory required to run a recipe depends on a number of factors, e.g. the sky subtraction scheme, number of files, and sky coverage. Below is a table with the minimum resident memory needed to process data as a function of recipe and number of input science frames. This table assumes that all recipe parameters are set to their default and a small maximum jitter offset between frames. In general, the execution time of a recipe can be shortened by choosing 'fast' stacking, but this will increase the memory requirements. The reason that `hawki_science_process` with 20 files uses more memory than with 50 files is because (using default recipe parameters) the stacking method changes from 'fast' to 'slow' (see description of `stk_fast` and `stk_nfst` parameters in §9.1).

Recipe	# science frames	Min. RAM
hawki_standard_process	4	3.2 GB
hawki_science_process	5	6.6 GB
hawki_science_process	10	10.5 GB
hawki_science_process	20	17.3 GB
hawki_science_process	50	7.9 GB

Table 4.1: Minimum memory requirements for selected HAWK-I recipes

4.2 Installing the Software

The HAWK-I pipeline is distributed as a standard pipeline kit package and can be obtained from the ESO web pages at `http://www.eso.org/sci/software/pipelines`. In addition to the HAWK-I pipeline, the distributed package contains all dependencies needed for the installation, the tools to run the recipes, this manual, and the installer utility for the kit.

Using the installer, the HAWK-I pipeline can be installed on recent versions of any major Linux distribution, as well as Mac OS X.

To install the HAWK-I pipeline, unpack the kit in a temporary location, go to the top level directory of the unpacked distribution package and execute the installer script as shown in the following example:

Note: The installation script uses the compiler which is found first in the path. If more than one compiler is installed on the system, make sure that your preferred compiler will be found first when the installation script is executed.

```
1> tar -zxf hawki-kit-X.Y.Z.tar.gz
```

```
2> cd hawki-kit-X.Y.Z
3> ./install_pipeline
```

Then follow the instructions on the screen. Once the script finishes successfully and the path variables have been set, the installation is complete.

4.3 EsoRex, Reflex, and Gasgano

ESO offers three different tools to process data obtained with VLT instruments: one command line tool (*EsoRex*), and two GUI based tools (*Gasgano* and *Reflex*). This manual describes how to use the pipeline with *EsoRex*; this option offers the most control on the reduction process.

Gasgano is useful as file browser for exploring HAWK-I data sets, however it is not recommended as an application to run the recipes. Using *Gasgano* to run HAWK-I recipes is not discussed further in this manual.

Reflex and the HAWK-I workflow provide a convenient way to execute a fixed setup of the reduction chain; it includes automatic data organization and processing. See [3] and [8] for details on how to use the HAWK-I Reflex workflow and Reflex itself.

5 Data Description

HAWK-I pipeline data can be separated into four general categories: *raw* files, *static calibration* files, *catalogue* files, and *product* files. Raw files are the unprocessed output of the HAWK-I instrument. Static calibration files are a set of mandatory and optional files for various calibration purposes. The catalogue files are optional data used to calibrate photometry and astrometry. Product files are the output of the HAWK-I pipeline processing (as reduced images, master calibration files, object catalogues, etc.). All general categories of files are described in this section.

In preparation for (and during) pipeline processing, all HAWK-I data must be classified into specific categories (e.g. DARK, FLAT, etc.) These files must then be associated with each other and with one (or more) recipes. This classification and association is done using FITS header keywords. *Gasgano* and *Reflex* can do the classification automatically by using the specially crafted file describing the classification and association rules (provided with the pipeline); the preferred method is to use *Reflex*. If using *EsoRex*, the user must classify and associate files themselves. Note that for large collections of files this can be tedious; the ESO convention on filenames also make this process prone to error. See §6.2.2 for details on how the classification and association is done.

5.1 Raw Data

Each HAWK-I raw FITS file contains a primary unit (without data), and 4 extensions (one for each of the 4 chips). Each extension is a 2048 by 2048 image containing data from a single chip; the EXTNAME keyword identifies the chip name (e.g. `CHIP3.INT1` for chip number 3). Beware that **the extension number in the raw files does not necessarily match the chip number**. For example, extension 1 contains data from chip 1, extension 2 contains data from chip 2, extension 3 contains data from chip 4 and extension 4 contains data from chip 3. The same ordering of the raw files is maintained in the pipeline product files.

Raw HAWK-I data retrieved from the ESO archive have the standard ESO archive file names, i.e. an instrument identifier followed by a time stamp. The time stamp corresponds to the contents of the FITS header keywords `MJD-OBS` and `DATE-OBS` respectively, i.e. to the date and time when the exposure was taken (a difference of 1 ms between the file name and the contents of the keywords may be present).

The files returned by the archive are **compressed** using the *fpack* utility; this is indicated by the file name suffix `.fits.fz` instead of the regular `.fits` suffix. The raw data files included in the pipeline kit (demonstration data) are also compressed. However, they have the `.fits` suffix in order for them to be recognised by the `Reflex` environment. These "tile compressed" files may be unpacked using the *funpack* tool distributed as part of the CFITSIO package [1]. There is no need to uncompress the raw data if the recipes are run either from the command line using *EsoRex* or inside the *Reflex* environment.

5.2 Static Calibration Data

The HAWK-I pipeline kit comes with a number of files that are used as part of the calibration of raw data. These files are referred to as "static" calibration files in order to distinguish them from output of recipes that are also used for calibration. Note that the "static" files that come with a particular version of the pipeline may differ

[1] http://heasarc.nasa.gov/fitsio/fitsio.html

from other versions of the pipeline. Moreover, as set of reference files (prefix `REF_`) are only part of the reflex demo-data and not of the pipeline kit itself.

A short summary of each of these files is given below. A more detailed description can be found in §A.1.

`hawki_phot.fits`: a binary FITS table containing quantities used to convert the HAWK-I photometric system to/from a reference system (e.g. 2MASS). There is one extension per system, and one row in each extension for each HAWK-I filter.
The UK in-kind HAWK-I archive reprocessing and new ESO HAWK-I imaging pipeline use the 2MASS catalogue to astrometrically and photometrically calibrate the science fields. The colour transformations used for the conversion between the HAWK-I photometric system and the system used for photometric calibration (e.g. 2MASS) can be found in the fits tables included in the data distribution (`HIERARCH ESO PRO CATG = PHOTCAL_TAB`).
For the current HAWK-I detectors and filters, the transformations to the 2MASS filters are:
$Y = 1.52 * J_{2MASS} - 0.52 * H_{2MASS}$
$J = 0.85 * J_{2MASS} + 0.15 * H_{2MASS}$
$H = 0.06 * J_{2MASS} + 0.94 * H_{2MASS}$
$Ks = 0.03 * J_{2MASS} + 0.97 * K_{2MASS}$
$CH4 = 0.06 * J_{2MASS} + 0.94 * H_{2MASS}$
$H2 = 0.03 * J_{2MASS} + 0.97 * K_{2MASS}$
$BrG = 0.03 * J_{2MASS} + 0.97 * K_{2MASS}$
$NB0984 = 1.62 * J_{2MASS} - 0.62 * H_{2MASS}$
$NB1060 = 1.50 * J_{2MASS} - 0.50 * H_{2MASS}$
$NB1190 = 0.85 * J_{2MASS} + 0.15 * H_{2MASS}$
$NB2090 = 0.03 * J_{2MASS} + 0.97 * K_{2MASS}$

`readgain.fits`: a binary FITS table containing typical values for the read noise and gain of each HAWK-I detector. These values may be calculated explicitly from any set of data using the `hawki_detector_noise` recipe; those values can then be used in subsequent recipes in the data reduction cascade. If a user chooses not to run `hawki_detector_noise`, the typical values in this file will be used by other recipes. This file was generated using `hawki_detector_noise` with data from March 2012 and contain values close to the median values derived from all data taken since HAWK-I began taking data.

`REF_DARK_XXX_YYY.fits`: a multi-extension FITS file containing a *reference* dark image for each HAWK-I detector. The values of `XXX` and `YYY` refer to the `DET.DIT` and `DET.NDIT` of the reference image, respectively. These files are used by the `hawki_dark_combine` recipe to compare the recipe product dark image ('master' dark) to a reference dark image. Only a selection of seven common DIT/NDIT pairs are provided with the detector in non-destructive read-out mode. The reference files were created using `hawki_dark_combine` with data from a range of dates; the provenance of each file can be seen in the primary headers. After the recipe created these products, the `update_procatg.py` script was used to change the `PRO.CATG` value to `REFERENCE_DARK` so that these files can be used by Reflex. The statistical distribution of pixels for the reference files are near the median values for all darks taken since HAWK-I began taking data.

`REF_FLAT_XXX.fits`: a multi-extension FITS file containing a *reference* twilight flat image for each HAWK-I detector. The values of `XXX` refer to the filter used for the reference image (`INS.FILT1.NAME`

or INS.FILT2.NAME). These files are used by the hawki_twilight_flat_combine recipe to compare the recipe product flat image ('master' flat) to a reference flat image. A selection of flats from nine filter are provided with the detector in non-destructive read-out mode. The reference files were created using hawki_twilight_flat_combine with data from May 2015; the provenance of each file can be seen in the primary headers. After the recipe created these products, the update_procatg.py script was used to change the PRO.CATG value to REFERENCE_TWILIGHT_FLAT so that the files can be used by Reflex.

SFD_dust_4096_ngp.fits: a FITS image containing the reddening values E_{B-V} in the northern Galactic hemisphere. These values are derived from the dust map described in [7]. The file is used by some HAWK-I recipes for photometric calibration purposes.

SFD_dust_4096_sgp.fits: as above, but for the southern Galactic hemisphere.

Note that the update_procatg.py script is part of the collection of static calibration files; this enables users to create their own reference files.

5.3 Photometric and Astrometric Catalogues

The HAWK-I pipeline can use a number of optional files to calibrate the astrometry and photometry of processed images. These files are only necessary if a user wishes to process data without an internet connection. If a user has an internet connection, the HAWK-I recipes can retrieve the required data automatically through the Strasbourg astronomical Data Center[2] (CDS). It is recommended that HAWK-I data use the 2MASS point source catalogue [2] for photometric calibration, and the WISE [1] catalogue for astrometric calibration.

A short summary of each of these files is given below; see §A.1 for further details.

index_casu_2mass_astrom.fits: a FITS binary table used by recipes as an index of the 2MASS point sources for astrometry. This file has a PRO.CATG keyword with value 'MASTER_2MASS_CATALOGUE_ASTROM'.

index_casu_2mass_photom.fits: a FITS binary table used by recipes as an index of the 2MASS point sources for photometry. This file has a PRO.CATG keyword with value 'MASTER_2MASS_CATALOGUE_PHOTOM'.

npscXXX.fits: a FITS binary table containing information about the 2MASS point sources. There is one file for each degree of right ascension, i.e. npsc000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$. These files must reside in the same directory as the 'MASTER_2MASS_CATALOGUE' file in order to work in combination with the pipeline.

index_casu_ppmxl_astrom.fits: a FITS binary table used by recipes as an index of the PPMXL point sources for astrometry. This file has a PRO.CATG keyword with value 'MASTER_PPMXL_CATALOGUE_ASTROM'.

[2] http://cdsweb.u-strasbg.fr/

`index_casu_ppmxl_photom.fits`: a FITS binary table used by recipes as an index of the PPMXL point sources for photometry. This file has a `PRO.CATG` keyword with value `'MASTER_PPMXL_CATALOGUE_PHOTOM'`.

`nppmxlXXX.fits`: a FITS binary table containing information about the PPMXL point sources. There is one file for each degree of right ascension, i.e. npsc000.fits contains objects with $0 < \mathrm{RA} \leq 1.0$. These files must reside in the same directory as the `'MASTER_PPMXL_CATALOGUE'` file in order to work in combination with the pipeline.

5.4 Pipeline Products

A brief description of the naming and content of pipeline products is given below; see §6 and §8 for more details.

5.4.1 Naming convention for files

The standard method for naming of a data product file for an ESO pipeline is for the product to be given a 'predictable' name. For example, two runs of the same recipe should yield products with exactly the same names. For recipes that generate several products of the same type, each file will have the same root name, but will have a number appended so that each file name is unique. This is the default naming convention for all the recipes in this package.

For those who would like a more descriptive file name, many of the recipes offer a command line switch called `prettynames`. This system works by creating the output file name from the input file name with an added suffix which denotes the type of product. For example, the recipe `hawki_science_process` produces calibrated exposure files and stacked exposures plus variance images for both. A SoF would include, amongst other things, a list of raw exposures. If the first in the list was called `HAWKI.2011-12-25T02:20:16.502.fits`, then the calibrated exposure and its variance would be called `HAWKI.2011-12-25T02:20:16.502_ex.fits` and `HAWKI.2011-12-25T02:20:16.502_ex_var.fits`, respectively. Stacks are named after the first raw exposure in the SoF, hence this stack and its variance map would be called `HAWKI.2011-12-25T02:20:16.502_st.fits` and `HAWKI.2011-12-25T02:20:16.502_st_var.fits`, respectively.

The full lists the default file names and the `prettyname` suffixes for each product type are given in §8.

5.4.2 Images

All images are created as multi-extension FITS files. The primary HDU only contains keywords; there is no data array. The primary HDU keywords are derived from the raw images used to create it. The PHU keywords provide general information about the observation; the extension keywords are relevant to the particular detector. If a detector is flagged as 'dead', a 'dummy' data array will still be created for the appropriate FITS extension, i.e. an image product will always have 4 extensions. Variance maps are also created for each calibrated image; see §9.2 for details.

5.4.3 Catalogues

There are two kinds of catalogues created by HAWK-I recipes: *object* catalogues and *matched standard* catalogues.

Object catalogues are FITS binary tables that contain information about statistically significant sources of emission detected in a recipe product image.

Matched standards catalogues are FITS binary tables that contain information about objects in a photometric or astrometric catalogue (§5.3) that were matched to objects detected in a recipe product image. The objects that appear in a matched standard catalogue are those that *may* be used to calibrate the image. For photometric calibration, the only objects that are actually used are those that satisfy the condition set by the `magerrcut` recipe parameter (see §6.5 & §6.6).

5.4.4 Diagnostic tables

Some recipes create FITS binary tables that provide a statistical description of the difference between a master calibration image and a reference image.

6 Data Reduction

6.1 Getting Started with EsoRex

EsoRex is a command-line tool used to execute the HAWK-I recipes (or any other standard VLT/VLTI instrument pipeline). Assuming *EsoRex* is in your `$PATH`, the general structure of an *EsoRex* command line is

```
1> esorex [esorex_options] [recipe [recipe_options] [sof1 [sof2]...]]
```

where `recipe` is the name of a recipe and `sof1, sof2, ...` are the names of files containing a set of frames (see below). Note that a user may point `EsoRex` to the location of recipe libraries by defining the `$ESOREX_PLUGIN_DIR` environment variable or by using the `recipe-dir` command line option. See [4] for a complete description of *EsoRex*.

A full list of *EsoRex* options can be listed with the command

```
1> esorex --help
```

The names and current values of recipe parameters are shown with the command

```
1> esorex --help <recipe name>
```

or

```
1> esorex --man-page <recipe name>
```

Note that the current parameter value can be set from a configuration file or on the command line. The default parameter value used by the recipe is shown between square brackets at the end of the one-line description of the parameter.

A full list of recipes that can be executed with *EsoRex* is shown with the command

```
1> esorex --recipes
```

The last argument of an *EsoRex* command is the name of one (or more) "set-of-frames" or "SoF" file(s). A SoF file is a text file comprising a list of input files for the recipe along with a one-word description of that file. Each line is composed of a `FILENAME`, followed by whitespace, followed by a `TAG`. The `TAG` tells the recipe what kind of file it is and how it should be used in the recipe (e.g. `BIAS, OBJECT`). Blank lines and those beginning with the '`#`' character are ignored. If several SoF files are provided, the recipe will concatenate the files before execution. If the number of files with the same tag exceeds the number accepted by the recipe, only the first files that appear in the concatenated SoF are used by the recipe. It is strongly recommended that absolute paths to files be provided rather than relative paths. However, *EsoRex* will recognise environment variables (e.g. `$RAW_DATA_DIR`).

An example of a valid set-of-frames is:

```
1> cat dark.sof
/data/hawki/raw/HAWKI.2014-02-21T09:48:53.153.fits DARK
$RAW_DATA_DIR/HAWKI.2014-02-21T09:50:36.645.fits DARK
#$RAW_DATA_DIR/HAWKI.2014-02-21T09:55:04.515.fits DARK
```

6.2 Data Organization, Classification, & Association

Running the HAWK-I pipeline recipes using *EsoRex* requires the user to construct the appropriate set-of-frames file(s). There are three steps to this process:

classify each input file with the correct `TAG` to be used by each recipe.

organize a set of input files for each recipe, e.g. a group of science observations taken as part of the same template, calibration files taken with the same filter as the science observations, etc.

associate the output of some recipes as input to other recipes.

6.2.1 Header Keywords

The following table summarizes selected FITS *primary* header keywords that are useful in classifying data files. The keywords are grouped by their context and intended use.

Primary Header Keyword Name	Description
Keywords for frame classification:	
`INSTRUME`	Name of the instrument, e.g. `HAWKI`
`DPR.CATG`	Raw data frame product category, e.g. `CALIB`
`DPR.TYPE`	Raw data frame product type, e.g. `DARK`
`DPR.TECH`	Raw data frame observation technique, e.g. `IMAGE`
`TPL.ID`	Name of template used to create raw frame, e.g. `HAWKI_img_cal_TwFlats`
`PRO.CATG`	Pipeline product category, e.g. `MEAN_SKY`
Keywords describing an observation:	
`OBJECT`	Target name
`RA`	Approximate telescope pointing RA (J2000) [deg]
`DEC`	Approximate telescope pointing DEC (J2000) [deg]
`MJD-OBS`	Modified Julian date near start of exposure
`DATE-OBS`	Human readable format of `MJD-OBS`
`OBS.NAME`	Name of the Observation Block (OB)
`OBS.START`	Start time of the OB in `DATE-OBS` format
`OBS.TARG.NAME`	Same as `OBJECT`
`TPL.START`	Start time of the template (within the OB)
`TPL.EXPNO`	Exposure sequence number within the template
`TPL.NEXP`	Number of exposures within the template

continued on next page

Primary Header Keyword Name	Description
Keywords describing the instrument/detector setup:	
DET.DIT	Detector integration time [sec]
DET.NDIT	Number of integrations, each with duration DET.DIT
DET.NCORRS.NAME	Name of detector read-out mode, e.g. "NonDest"
INS.FILT1.NAME	Name of filter in first filter position
INS.FILT2.NAME	Name of filter in second filter position

Almost all HAWK-I pipeline product files contain a number of Quality Control (QC) parameters. These QC parameters are values which are computed by the recipes as indicators of the quality of the raw data and the reduction process. They are available from the FITS header of the pipeline products as hierarchical keywords starting with the leading group component 'QC'.

6.2.2 Classification

As discussed in §6.1, each recipe must be provided with a *set-of-frames* (SoF) file or files. The SoF file(s) consist of a list of filenames and frame tags for each file. This section discusses the valid frame tags for each recipe and recommendations for how to determine the frame tag from header keywords. Recall that HAWK-I data fall into one of four categories: raw data (§5.1), static calibration data (§5.2), photometric/astrometric catalogues (§5.3), and pipeline product data (§5.4). The tables below describe the frame tags for each category in turn.

Table 6.1 shows the frame tags for HAWK-I raw data files and the recipes that accept files with those tags. It is recommended that these tags be determined from a unique combination of the primary header keywords DPR.CATG, DPR.TYPE, and DPR.TECH, as shown in the last three columns.

Frame tag	Recipe(s)	DPR.CATG	DPR.TYPE	DPR.TECH
DARK	hawki_dark_combine, hawki_detector_noise	CALIB	DARK	IMAGE
FLAT_TWILIGHT	hawki_twilight_flat_combine, hawki_detector_noise	CALIB	FLAT	IMAGE
STD	hawki_standard_process	CALIB [or TEST]	STD	IMAGE
OBJECT	hawki_science_process	SCIENCE	OBJECT	IMAGE
SKY	hawki_science_process	SCIENCE	SKY	IMAGE

Table 6.1: Frame tags for raw HAWK-I data

Table 6.2 shows the frame tags for HAWK-I static calibration files and the recipes that accept files with those tags. The third column shows the filename provided in the pipeline kit that is recommended to be identified with the tag; see §5.2 for a description of each file. A user is free to create her own static calibration file for any tag, but it must meet the data format specification outlined in §A.1. Note that some of these tags are *optional* inputs to the recipe; see §8 for details.

Frame tag	Recipe(s)	Pipeline kit filename
REFERENCE_DARK	hawki_dark_combine	REF_DARK_XXX_XXX.fits
REFERENCE_TWILIGHT_FLAT	hawki_twilight_flat_combine	REF_FLAT_XXX_XXX.fits
MASTER_READGAIN	hawki_standard_process, hawki_science_process	readgain.fits
PHOTCAL_TAB	hawki_standard_process, hawki_science_process, hawki_science_postprocess	hawki_phot.fits
SCHLEGEL_MAP_NORTH	hawki_standard_process, hawki_science_process, hawki_science_postprocess	SFD_dust_4096_ngp.fits
SCHLEGEL_MAP_SOUTH	hawki_standard_process, hawki_science_process, hawki_science_postprocess	SFD_dust_4096_sgp.fits

Table 6.2: Frame tags for HAWK-I static calibration data

Table 6.3 shows the frame tags for HAWK-I photometric and astrometric catalogue index files and the recipes that accept files with those tags. As stated before, by default, the catalogue fields needed to calibrate the images are retrieved by the pipeline from the CDS. The third column shows the filename provided in the pipeline kit that is recommended to be identified with the tag; see §5.2 for a description of each file; a blank filename indicates that such a file should be produced by the user. A user is free to create her own index file for any tag, but it must meet the data format specification outlined in §A.2. The MASTER_LOCAL_CATALOGUE_ASTROM and MASTER_LOCAL_CATALOGUE_PHOTOM tags allow a user to use their own photometric or astrometric catalogue to calibrate their data, respectively (no files with this tag are provided with the pipeline kit). Note that some of these tags are *optional* inputs to the recipe; see §8 for details.

Frame tag	Recipe(s)	Pipeline kit filename
MASTER_2MASS_CATALOGUE_ASTROM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	index_casu_2mass_astrom.fits
MASTER_2MASS_CATALOGUE_PHOTOM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	index_casu_2mass_photom.fits
MASTER_PPMXL_CATALOGUE_ASTROM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	index_casu_ppmxl_astrom.fits
MASTER_PPMXL_CATALOGUE_PHOTOM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	index_casu_ppmxl_photom.fits
MASTER_LOCAL_CATALOGUE_ASTROM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	

continued on next page

Frame tag	Recipe(s)	Pipeline kit filename
MASTER_LOCAL_CATALOGUE_PHOTOM	hawki_standard_process, hawki_science_process, hawki_science_postprocess	

Table 6.3: Frame tags for index files to HAWK-I photometric and astrometric catalogue

Table 6.4 lists the input frame tags accepted by each HAWK-I recipe. Note that the same tag may be accepted by more than one recipe. Optional input frames are enclosed by square brackets. More details about the required and optional input frame tags for each HAWK-I recipe can be found in the recipe reference chapter (§8).

Recipe	Frame tag
hawki_dark_combine	DARK, [REFERENCE_DARK], [MASTER_BPM], [MASTER_CONF]
hawki_twilight_flat_combine	FLAT_TWILIGHT, MASTER_DARK, [REFERENCE_TWILIGHT_FLAT], [MASTER_BPM], [MASTER_CONF]
hawki_detector_noise	DARK, FLAT_TWILIGHT, [MASTER_BPM], [MASTER_CONF]
hawki_standard_process	STD, MASTER_DARK, MASTER_TWILIGHT_FLAT, PHOTCAL_TAB, MASTER_CONF, MASTER_READGAIN, SCHLEGEL_MAP_NORTH, SCHLEGEL_MAP_SOUTH, [MASTER_2MASS_CATALOGUE_ASTROM], [MASTER_PPMXL_CATALOGUE_ASTROM], [MASTER_LOCAL_CATALOGUE_ASTROM], [MASTER_2MASS_CATALOGUE_PHOTOM], [MASTER_PPMXL_CATALOGUE_PHOTOM], [MASTER_LOCAL_CATALOGUE_PHOTOM]
hawki_science_process	OBJECT, [SKY], MASTER_DARK, MASTER_TWILIGHT_FLAT, [MASTER_SKY], [MASTER_SKY_VAR], PHOTCAL_TAB, MASTER_CONF, MASTER_READGAIN, SCHLEGEL_MAP_NORTH, SCHLEGEL_MAP_SOUTH, [MASTER_2MASS_CATALOGUE_ASTROM], [MASTER_PPMXL_CATALOGUE_ASTROM], [MASTER_LOCAL_CATALOGUE_ASTROM], [MASTER_2MASS_CATALOGUE_PHOTOM], [MASTER_PPMXL_CATALOGUE_PHOTOM], [MASTER_LOCAL_CATALOGUE_PHOTOM], [MATCHSTD_PHOTOM], [MASTER_OBJMASK]

continued on next page

Recipe	Frame tag
hawki_science_postprocess	BASIC_CALIBRATED_SCI, BASIC_VAR_MAP, PHOTCAL_TAB, MASTER_CONF, SCHLEGEL_MAP_NORTH, SCHLEGEL_MAP_SOUTH, [MASTER_2MASS_CATALOGUE_ASTROM], [MASTER_PPMXL_CATALOGUE_ASTROM], [MASTER_LOCAL_CATALOGUE_ASTROM], [MASTER_2MASS_CATALOGUE_PHOTOM], [MASTER_PPMXL_CATALOGUE_PHOTOM], [MASTER_LOCAL_CATALOGUE_PHOTOM], [MATCHSTD_PHOTOM]

Table 6.4: Accepted frame tags for each HAWK-I recipe

6.2.3 Organization

It is important to ensure that the input files sent to a recipe represent a coherent group of data. The variation in users' observing strategies, science outcomes, etc. make it difficult to provide concrete rules for how to do this. It is recommended that calibration data and science data are grouped together by detector read-out mode (DET.NCORRS.NAME), integration time (DET.DIT and DET.NDIT), and filter (INS.FILT1.NAME and INS.FILT2.NAME). Science observations are generally grouped together by target name (OBS.TARG.NAME) and either template (TPL.START) or OB (OBS.START). The use of calibration data with different properties than the science data may yield less than ideal results.

6.2.4 Association

Care must also be taken in associating the output of some recipes with input to other recipes. There are many instances where an output file from one recipe may be used as an input files for another (or the same!) recipe. In this situation, the frame tag for the input files is (usually) identical to the PRO.CATG value of the output file. There are some exceptions to this: MASTER_SKY (from MEAN_SKY), MASTER_SKY_VAR (from MEAN_SKY_VAR). Table 6.5 lists PRO.CATG keyword values for all output files that may be used as an input file for a HAWK-I recipe. The lefthand column is the name of the recipe that generates a file with the PRO.CATG value(s) in the right hand column. Note that this table is not a complete list of all outputs from all of the recipes.

Recipe	Output PRO.CATG
hawki_dark_combine	MASTER_DARK
hawki_twilight_flat_combine	MASTER_TWILIGHT_FLAT, MASTER_CONF, MASTER_BPM

continued on next page

Recipe	**Output** PRO.CATG
hawki_detector_noise	MASTER_READGAIN
hawki_science_process	BASIC_CALIBRATED_SCI, BASIC_VAR_MAP, MATCHSTD_PHOT, MEAN_SKY, MEAN_SKY_VAR

Table 6.5: Output pipeline products that may be used as input frames

The following is a recommendation for how to associate files for each recipe. This description is implemented in the OCA rules found in the HAWK-I Reflex workflow and assumes default values for recipe parameters. Please note that this may not be appropriate for a user's particular data and/or calibration plan, e.g. using a MASTER_SKY file and/or a MATCHSTD_PHOTOM table as input.

hawki_dark_combine: Group raw dark frames together that share the same DET.NCORRS.NAME, DET.DIT, DET.NDIT, and TPL.START keywords. Optionally select a REFERENCE_DARK that shares the same DET.NCORRS.NAME, DET.DIT, and DET.NDIT keywords for each group. Process the data to create a MASTER_DARK file for each group.

hawki_twilight_flat_combine: Group raw twilight frames together that share the same DET.NCORRS.NAME, DET.DIT, DET.NDIT, INS.FILT1.NAME, INS.FILT2.NAME, and OBS.START keywords. Select a MASTER_DARK that shares the same DET.DIT keyword for each group. Optionally select a REFERENCE_TWILIGHT_FLAT that shares the same DET.NCORRS.NAME, DET.DIT, DET.NDIT, INS.FILT1.NAME, and INS.FILT2.NAME keywords for each group. Process the data to create MASTER_TWILIGHT_FLAT, MASTER_CONF, and MASTER_BPM files for each group.

hawki_detector_noise: Select two raw dark frames together that have DET.NDIT equal to 1 and share the same DET.NCORRS.NAME and DET.DIT keywords. Select two raw twilight flat frames together that have the same DET.NCORRS.NAME keyword as the dark frames and share the same INS.FILT1.NAME and INS.FILT2.NAME keywords. Optionally select a MASTER_CONF file generated from a list of raw twilight frames that contains the two raw flat frames referred to in the previous sentence. Process the data to create a MASTER_READGAIN file for each group.

hawki_standard_process: Group raw STD frames together that share the same DET.NCORRS.NAME, DET.DIT, DET.NDIT, INS.FILT1.NAME, INS.FILT2.NAME, OBS.TARG.NAME, and OBS.START keywords. Select a MASTER_DARK that shares the same DET.DIT keyword for each group. Select a MASTER_TWILIGHT_FLAT and MASTER_CONF that share the same DET.NCORRS.NAME, INS.FILT1.NAME, and INS.FILT2.NAME keyword for each group. Select a MASTER_READGAIN file. Select PHOTCAL_TAB, SCHLEGEL_MAP_NORTH and SCHLEGEL_MAP_SOUTH files from the static calibration directory. Optionally select astrometric and/or photometric catalogue index files (if not using CDS). Process the data to create a number of data products.

hawki_science_process: Group raw science frames together that share the same DET.NCORRS.NAME, DET.DIT, DET.NDIT, INS.FILT1.NAME, INS.FILT2.NAME, and OBS.START keywords. Select a MASTER_DARK that shares the same DET.DIT keyword for each group. Select a

MASTER_TWILIGHT_FLAT and MASTER_CONF that share the same DET.NCORRS.NAME, INS.FILT1.NAME, and INS.FILT2.NAME keyword for each group. Select a MASTER_READGAIN file. Select PHOTCAL_TAB, SCHLEGEL_MAP_NORTH and SCHLEGEL_MAP_SOUTH files from the static calibration directory. Optionally select astrometric and/or photometric catalogue index files (if not using CDS). Process the data to create a number of data products including BASIC_CALIBRATED_SCI and BASIC_VAR_MAP files.

hawki_science_postprocess: Group together the BASIC_CALIBRATED_SCI and BASIC_VAR_MAP files from the output of one instance of hawki_science_process. Select the same MASTER_CONF file that was used in that instance of hawki_science_process. Select PHOTCAL_TAB, SCHLEGEL_MAP_NORTH and SCHLEGEL_MAP_SOUTH files from the static calibration directory. Optionally select astrometric and/or photometric catalogue index files (if not using CDS). Process the data to create a number of data products.

6.3 Data Reduction Cascade

Figures 6.1–6.3 show a visual representation and overview of the data reduction with the HAWK-I pipeline. The cascade for generating master calibration data is shown in Figure 6.1. Figure 6.2 shows the cascade for the recipe that processes standard star fields. The cascade for processing science frames is shown in Figure 6.3.

6.4 Calibration

This section describes the steps to generate the 'master' calibration data: a master dark, flat field, bad pixel map, confidence map, and detector noise and gain values. These steps should be taken in the order they appear in this manual. Note that the recipes described in this section are run with most recipe parameters set to their default values. Only a brief description of the recipes is provided here; see §8 for details.

The file names listed below can be found in the demo data as part of the pipeline kit. Note, however, that the absolute pathnames shown in the SoF files depend on where the user has installed her data; they may not be the same as shown below.

6.4.1 Dark

The first required step is to combine a set of raw dark frames into one 'master' dark image. The hawki_dark_combine recipe takes a list of DARK frames (with the same detector and exposure time parameters), and combines them to form a mean dark (after rejecting outlying pixel values).

If a REFERENCE_DARK file is supplied in the SoF file, then a difference image is formed between the REFERENCE_DARK and the MASTER_DARK. This difference image (DIFFIMG_DARK) can be useful for looking at the evolution of the structure of the dark current and the reset anomaly over time. It can also be useful as a quick check that the recipe has created a reasonably valid master dark. The REFERENCE_DARK should have been taken with the detector settings as the MASTER_DARK (e.g., readout mode, DIT, NDIT).

The median value of the difference image and the root-mean-square deviation within several small areas on the chip are written to a 'difference image statistics table' (DIFFIMG_STATS_DARK). The inclusion of either

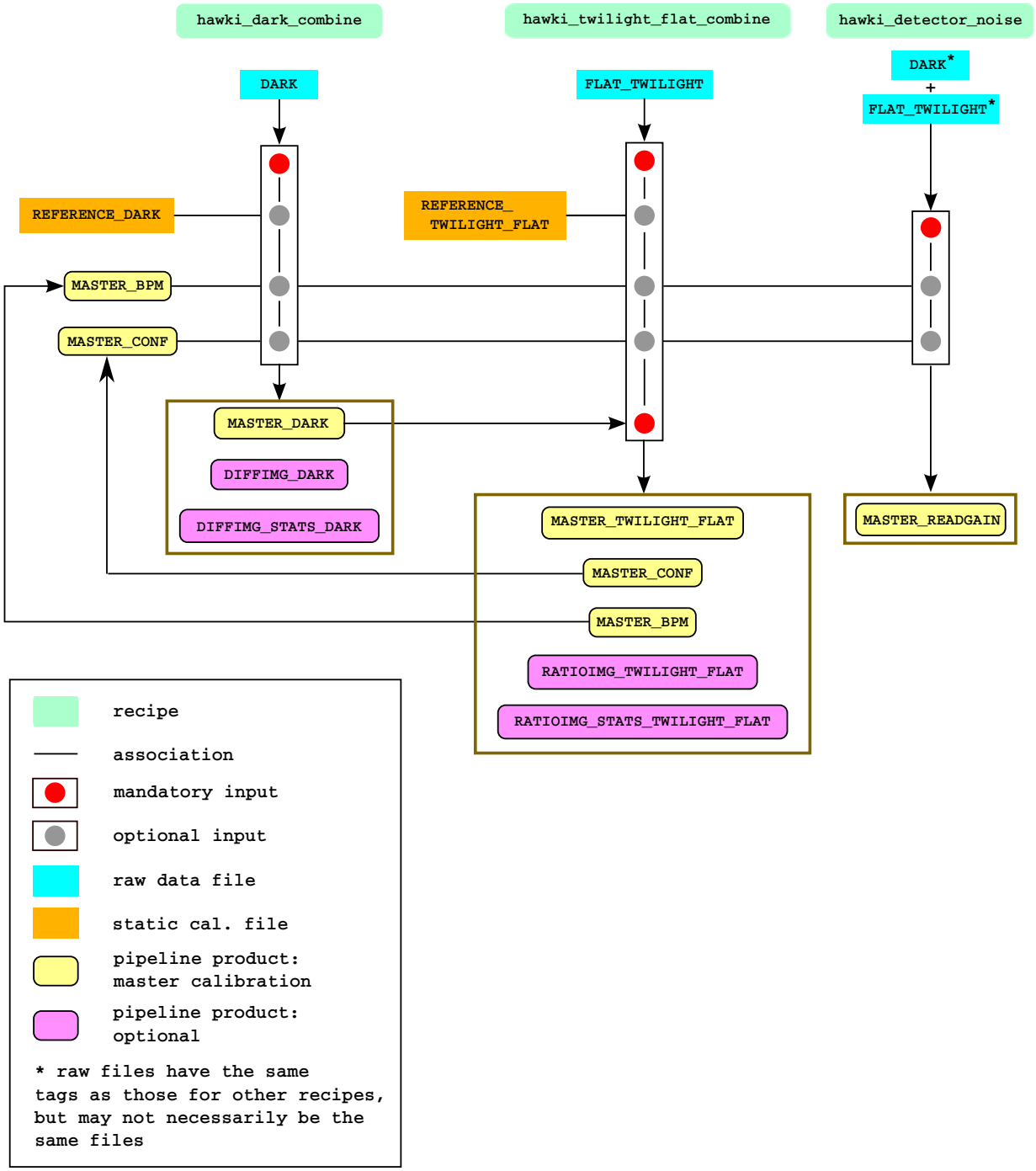

Figure 6.1: Data reduction cascade and tag association for HAWK-I master calibration recipes.

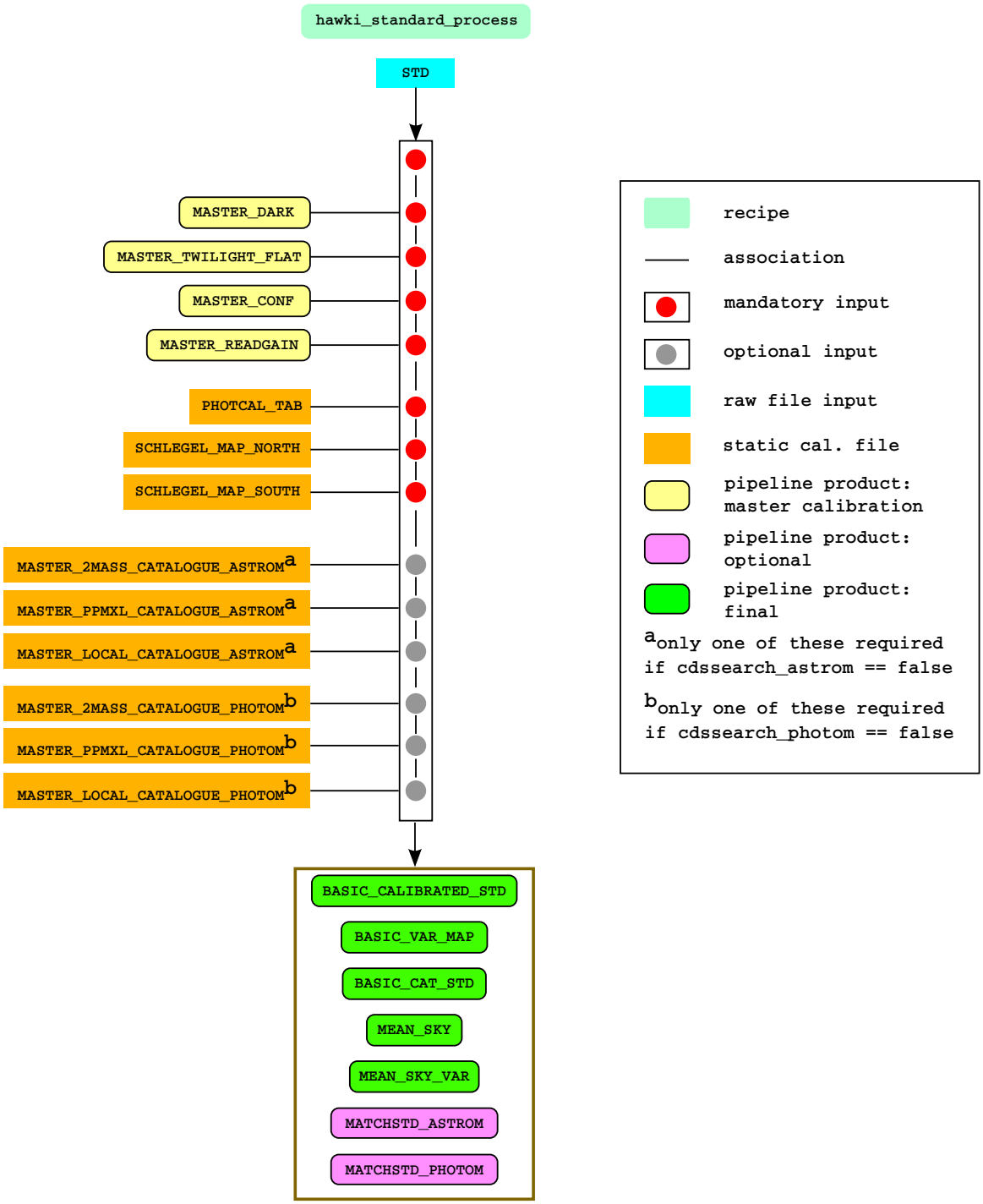

Figure 6.2: Data reduction cascade and tag association for the HAWK-I standard star recipe.

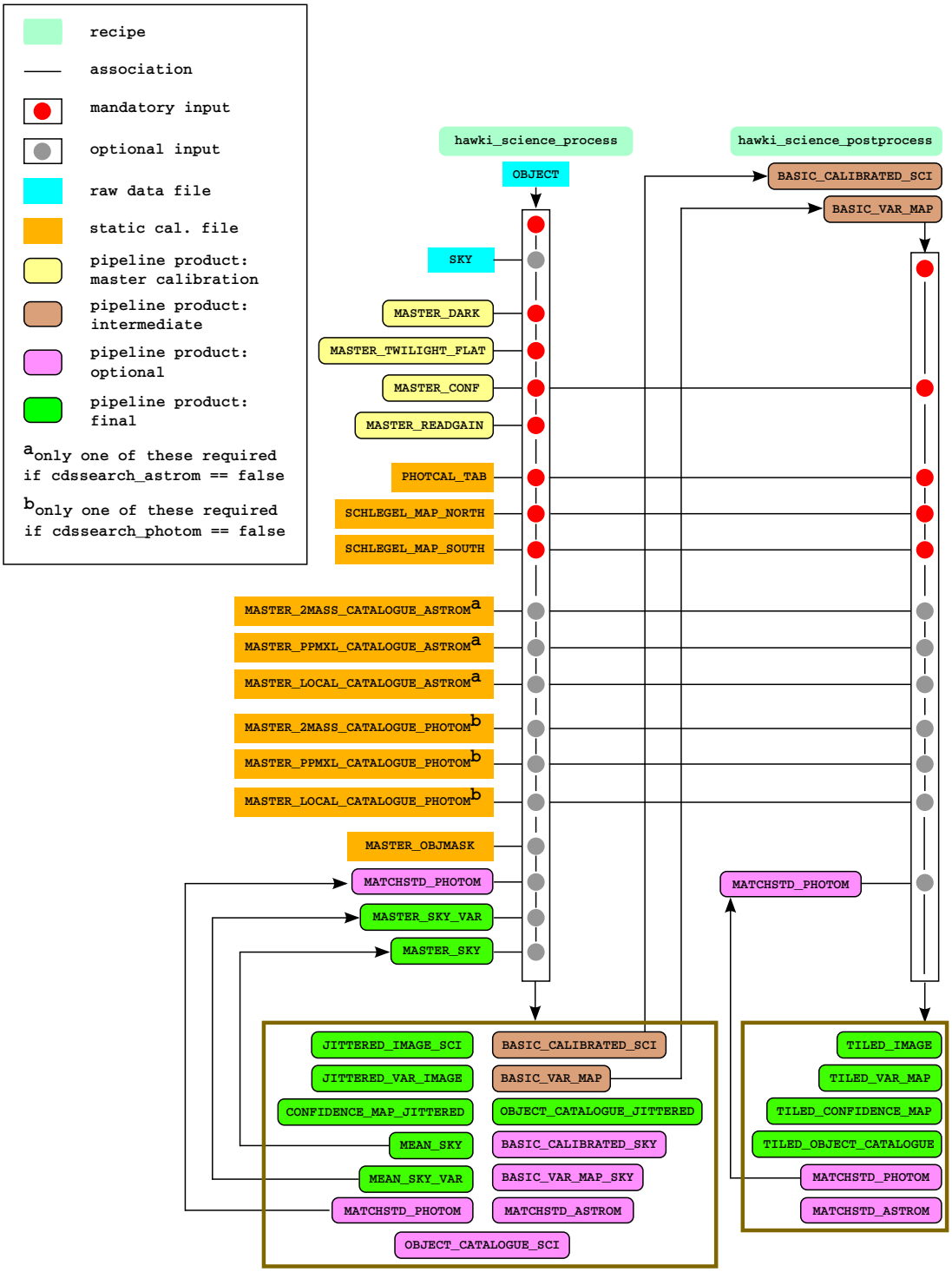

Figure 6.3: Data reduction cascade and tag association for HAWK-I science recipes.

a master confidence map (MASTER_CONF) or a bad pixel mask (MASTER_BPM) in the SoF file can aid in masking out bad pixels when evaluating the statistics that appear in DIFFIMG_STATS_DARK. If neither a MASTER_CONF nor MASTER_BPM are provided, the statistics are calculated assuming that all pixels in the difference image are valid.

For example, in order to make a master dark image from the demo data where DIT = 2 sec, NDIT = 1, the SoF file dark.sof should contain these files:

```
1> cat dark.sof
$RAW_DATA_DIR/HAWKI.2008-11-18T09:25:12.646.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:25:22.242.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:25:31.881.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:25:41.512.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:25:51.090.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:26:00.666.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:26:10.295.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:26:19.919.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:26:29.561.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:26:39.145.fits DARK
$STATIC_CALIB_DATA_DIR/REF_DARK_2_1.fits REFERENCE_DARK
```

where the environmental variables ('$' prefix) refer to the location of the user's data. The command to create the master dark using the default recipe input parameters is

```
1> esorex hawki_dark_combine dark.sof
```

This will create three files:

darkcomb.fits: the MASTER_DARK pipeline product file.

darkdiff.fits: the DIFFIMG_DARK pipeline product file.

darkdifftab.fits: the DIFFIMG_STATS_DARK pipeline product file.

and potentially an esorex.log file, depending on the user's esorex settings. Note that the filenames will be different from those above if the parameter prettynames is set to 'true' (see §8). If a REFERENCE_DARK is omitted from the SoF file, the darkdiff.fits and darkdifftab.fits files will not be created. A screenshot of each of these pipeline product files is shown in Figures6.4 - 6.6.

6.4.2 Flat Field

The second required step is to combine a set of raw twilight flat frames into one 'master' flat image. The hawki_twilight_flat_combine recipe takes a list of FLAT_TWILIGHT frames (with the same detector and filter parameters), corrects each frame for dark current, and combines them to form a normalised, mean

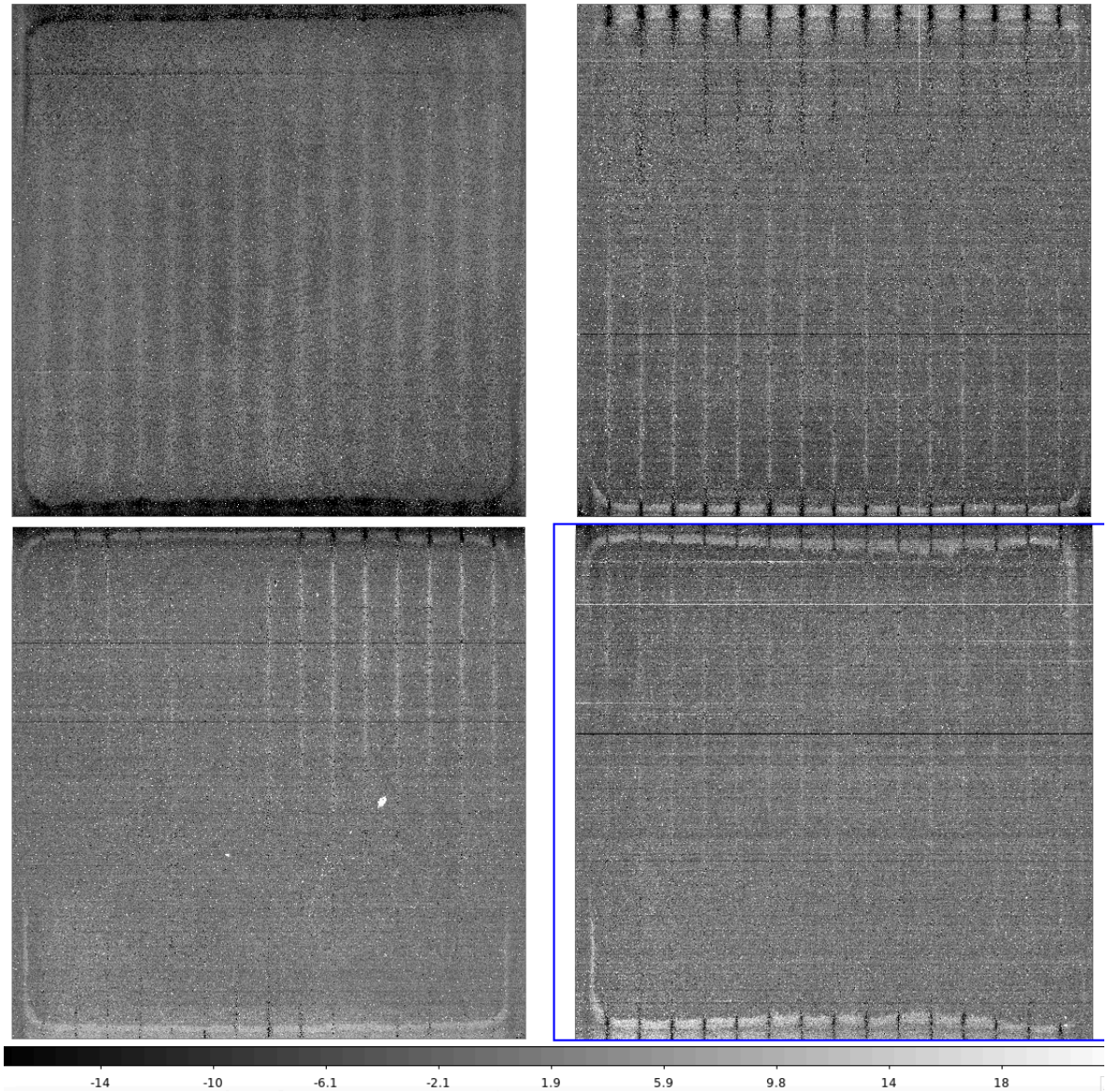

Figure 6.4: Example HAWK-I master dark for DIT = 2 sec, NDIT = 1. Each quadrant of the figure shows a different chip.

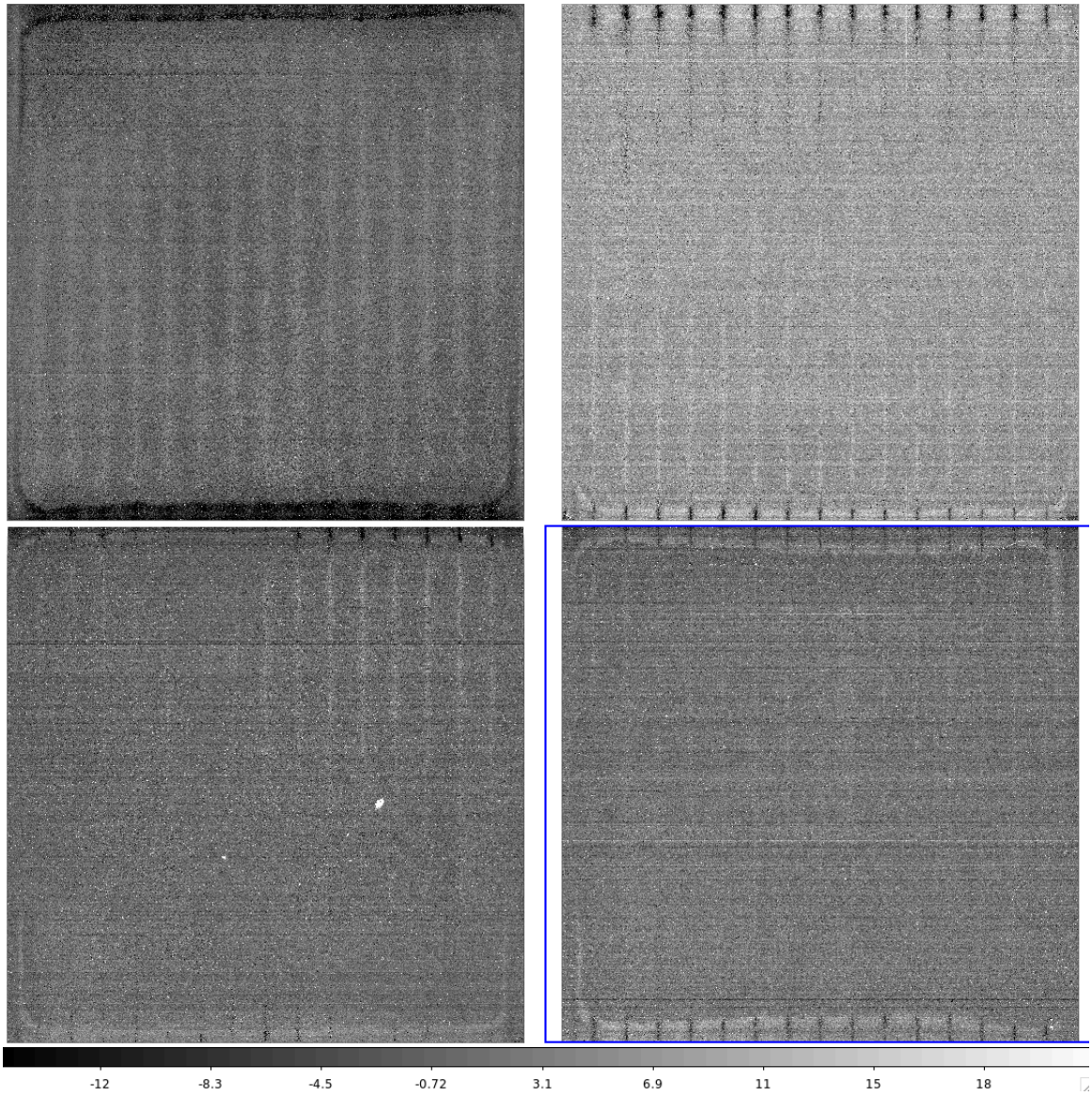

Figure 6.5: Example HAWK-I DIFFIMG_DARK, i.e. the difference between a master dark and a reference dark.

Select All Invert	xmin 1J pixels Modify	xmax 1J pixels Modify	ymin 1J pixels Modify	ymax 1J pixels Modify	chan 1J pixels Modify	mean 1E ADU Modify	median 1E ADU Modify	variance 1E ADU**2 Modify	mad 1E ADU Modify
1	1	128	1	256	1	3.344481E+00	1.880642E+00	1.002614E+04	7.657986E+00
2	1	128	257	512	1	9.705391E+00	8.806858E+00	5.952076E+03	6.239584E+00
3	1	128	513	768	1	1.094852E+01	1.037457E+01	6.877982E+03	5.484809E+00
4	1	128	769	1024	1	1.035770E+01	1.015234E+01	5.567814E+03	5.158854E+00
5	1	128	1025	1280	1	1.013460E+01	8.935329E+00	8.514952E+03	5.006077E+00
6	1	128	1281	1536	1	8.488305E+00	7.427951E+00	6.454015E+03	5.737847E+00
7	1	128	1537	1792	1	4.209230E+00	4.215712E+00	1.368063E+03	5.946615E+00
8	1	128	1793	2048	1	5.758021E-01	9.045139E-01	2.440326E+03	6.291233E+00

Figure 6.6: Extract from example HAWK-I `DIFFIMG_STATS_DARK` table, i.e. a statistical description of `DIFFIMG_DARK`. The first eight rows from the first extension are shown, corresponding to the first channel on `CHIP1.INT`.

twilight flat (after rejecting outlying pixel values). The median value of a master twilight flat frame for each chip is 1.0. The recipe also creates a master bad pixel map and a master confidence map; these can be used to identify bad pixels. See §9 for a detailed description of the bad pixel maps and confidence maps.

Note that an additional required input to the recipe is a MASTER_DARK. This MASTER_DARK should have been created from raw data that share the same detector settings (e.g. readout mode, DIT, NDIT) as the raw twilight flat data.

If a REFERENCE_TWILIGHT_FLAT file is supplied in the SoF file, then a ratio image is formed between the REFERENCE_TWILIGHT_FLAT and the MASTER_TWILIGHT_FLAT. This ratio image (RATIOIMG_TWILIGHT_FLAT) can be useful for looking at the evolution of the structure of the filter or flat field over time. It can also be useful as a quick check that the recipe has created a reasonably valid master flat. The REFERENCE_TWILIGHT_FLAT should have been taken with the same filter as the MASTER_TWILIGHT_FLAT.

The median value of the ratio image and the root-mean-square deviation within several small areas on the chip are written to a 'difference image statistics table' (RATIOIMG_STATS_TWILIGHT_FLAT). The inclusion of either a master confidence map (MASTER_CONF) or a bad pixel mask (MASTER_BPM) in the SoF file can aid in masking out bad pixels when evaluating the statistics that appear in RATIOIMG_STATS_TWILIGHT_FLAT. If neither a MASTER_CONF nor MASTER_BPM are provided, the statistics are evaluated assuming that all pixels in the ratio image are valid. If both are provided, the MASTER_CONF will be used.

For example, in order to make a master flat image from the demo data with the Y band filter, the SoF file should contain these files:

```
1> cat flat.sof
$RAW_DATA_DIR/HAWKI.2008-11-17T23:35:31.179.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:35:51.113.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:36:11.052.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:36:29.549.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:36:49.442.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:37:09.370.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:37:29.305.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:37:49.234.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:38:09.179.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:38:29.066.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:38:49.005.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:39:08.893.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:39:28.793.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:39:48.682.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:40:08.623.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:40:28.510.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:40:48.405.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:41:08.293.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:41:28.235.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:41:48.157.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:42:08.062.fits FLAT_TWILIGHT
```

```
$RAW_DATA_DIR/HAWKI.2008-11-17T23:42:27.988.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:42:47.929.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:43:07.823.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:43:27.726.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:43:47.612.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:44:07.515.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:44:27.403.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:44:47.342.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:45:07.226.fits FLAT_TWILIGHT
$STATIC_CALIB_DATA_DIR/REF_FLAT_Y.fits REFERENCE_TWILIGHT_FLAT
$MASTER_CALIB_DATA_DIR/darkcomb_Y_flat.fits MASTER_DARK
```

where the environmental variables (`'$'` prefix) refer to the pathname of the user's data. Note that the `MASTER_DARK` frame here is **not the same file** as the one described in §6.4.1 above. The reason is that the raw twilight flat frames here were taken with DIT = 10 sec, NDIT = 1 instead of DIT = 2 sec, NDIT = 1. The `MASTER_DARK` appropriate for the creation of a master flat here should be created from the collection of raw dark frames in the demo data set with DIT = 10 sec, NDIT = 1.

The command to create the master flat using the default recipe input parameters is

```
1> esorex hawki_twilight_flat_combine flat.sof
```

This will create five files:

`twilightcomb.fits`: the `MASTER_TWILIGHT_FLAT` pipeline product file.

`twilightconf.fits`: the `MASTER_CONF` pipeline product file.

`bpmmap.fits`: the `MASTER_BPM` pipeline product file.

`twilightratio.fits`: the `RATIOIMG_TWILIGHT_FLAT` pipeline product file.

`twilightratiotab.fits`: the `RATIOIMG_STATS_TWILIGHT_FLAT` pipeline product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true' (see §8). If a `REFERENCE_TWILIGHT_FLAT` is omitted from the SoF file, the `twilightratio.fits` and `twilightratiotab.fits` files will not be created. Screenshots of some of these pipeline product files are shown in Figures6.7 - 6.9.

6.4.3 Detector Noise [optional]

In order to process standard star images or science images, a `MASTER_READGAIN` file is needed. This file is a table containing the readnoise, gain, and covariance for each HAWK-I chip. If a user chooses to use the `MASTER_READGAIN` provided with the pipeline kit, they should skip this section. However, if a user would like

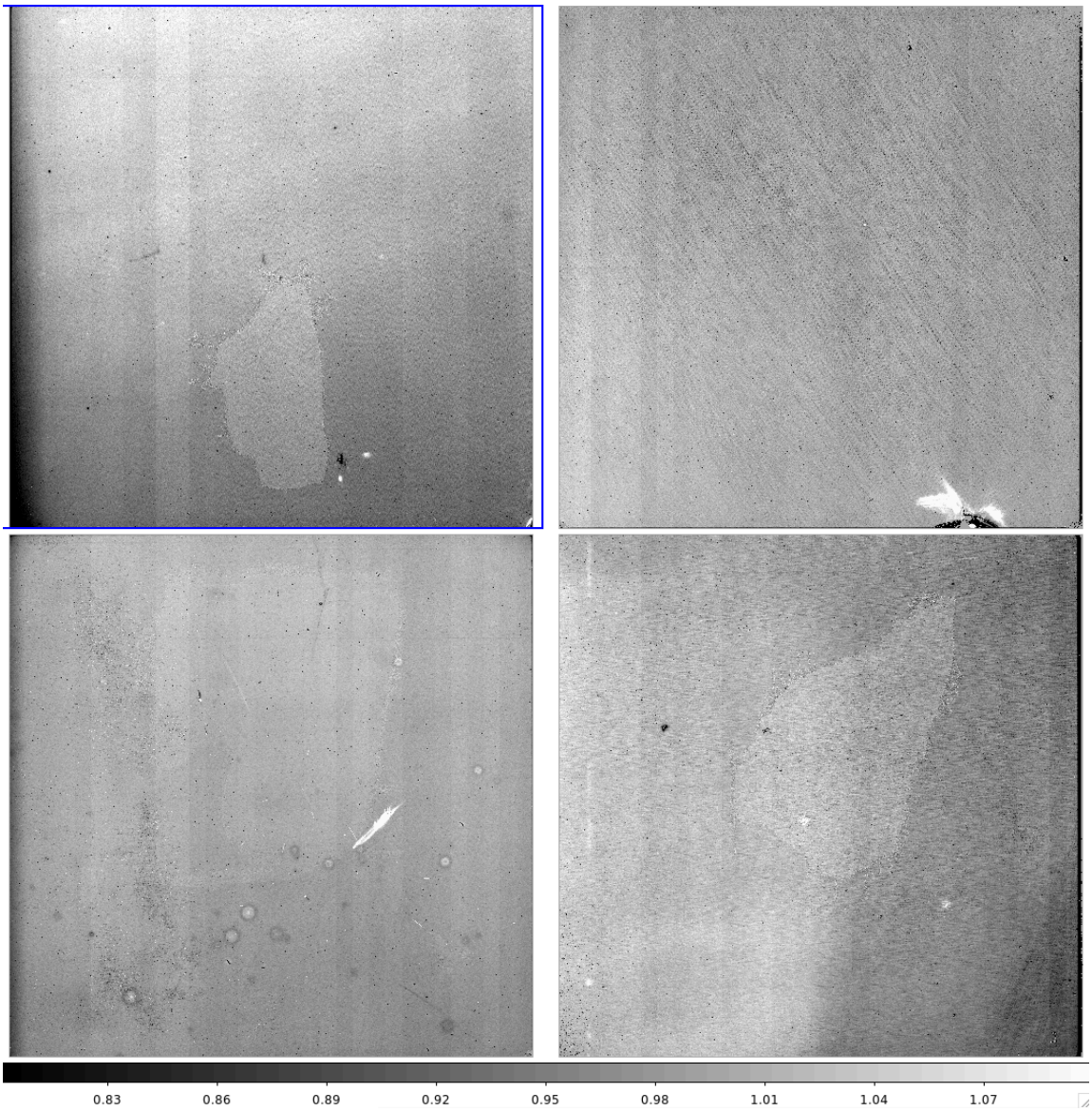

Figure 6.7: Example HAWK-I master twilight flat for filter Y. Each quadrant of the figure shows a different chip.

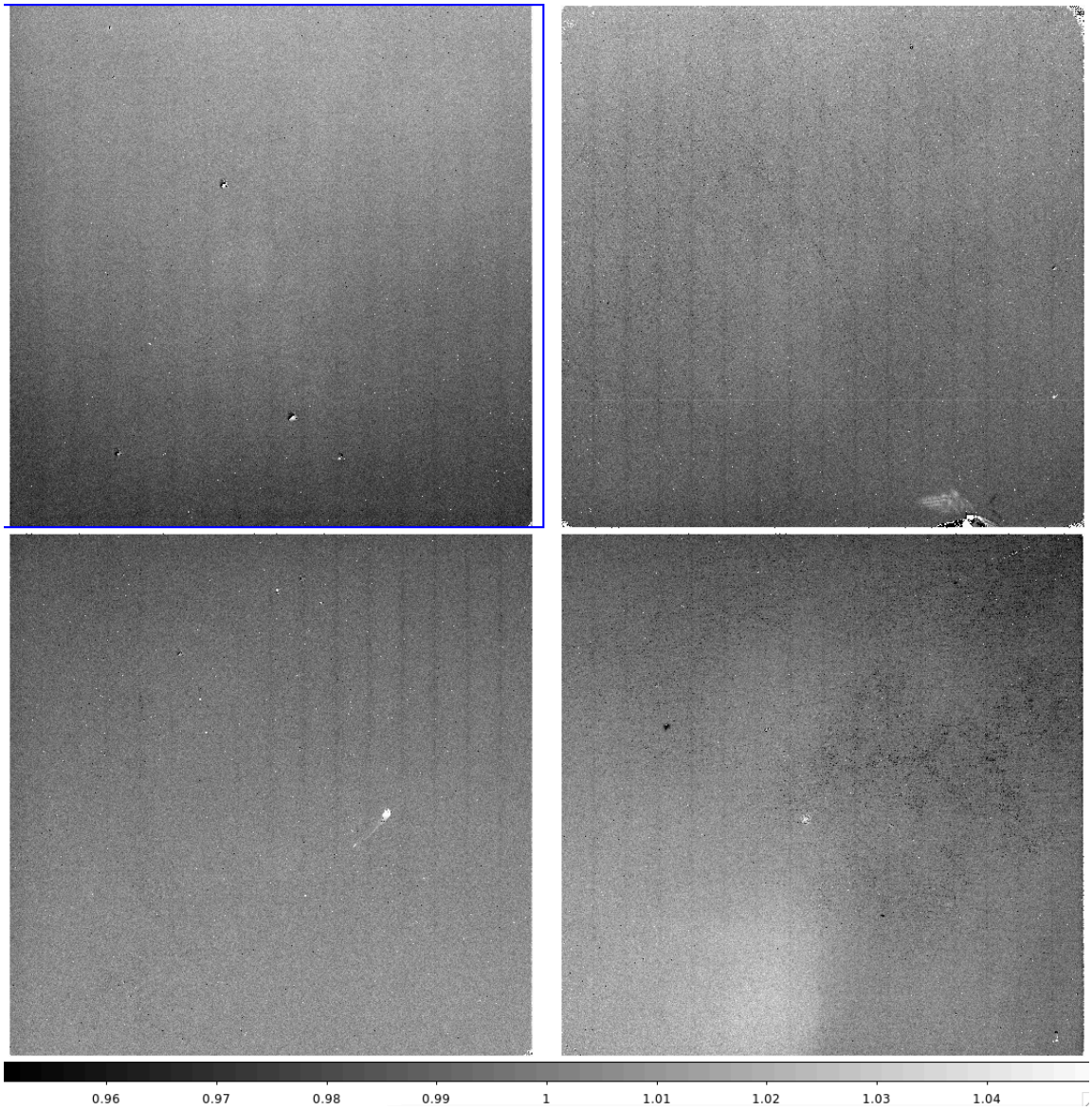

Figure 6.8: Example HAWK-I `RATIOIMG_TWILIGHT_FLAT`, i.e. the ratio of a master twilight flat and a reference twilight flat.

Select	xmin	xmax	ymin	ymax	chan	mean	median	variance	mad
All	1J	1J	1J	1J	1J	1E	1E	1E	1E
Invert	pixels	pixels	pixels	pixels	pixels	ADU	ADU	ADU**2	ADU
	Modify	Modify	Modify	Modify	Modify	Modify	Modify	Modify	Modify
1	1	128	1	256	1	9.752831E-01	9.753390E-01	3.846300E-05	3.852248E-03
2	1	128	257	512	1	9.784533E-01	9.784002E-01	3.501322E-05	3.812373E-03
3	1	128	513	768	1	9.843376E-01	9.843038E-01	3.480299E-05	3.854990E-03
4	1	128	769	1024	1	9.902707E-01	9.902628E-01	3.525537E-05	3.923416E-03
5	1	128	1025	1280	1	9.944050E-01	9.944449E-01	3.217696E-05	3.737867E-03
6	1	128	1281	1536	1	9.971778E-01	9.972218E-01	3.369492E-05	3.872931E-03
7	1	128	1537	1792	1	1.002183E+00	1.002188E+00	3.794593E-05	3.993869E-03
8	1	128	1793	2048	1	1.006065E+00	1.006061E+00	3.689777E-05	3.964365E-03

Figure 6.9: Extract from example HAWK-I `RATIOIMG_STATS_TWILIGHT_FLAT` table, i.e. a statistical description of `RATIOIMG_TWILIGHT_FLAT`. The first eight rows from the first extension of the file are shown, corresponding to the first channel on `CHIP1.INT`.

to create their own MASTER_READGAIN file from raw data, she should use the hawki_detector_noise recipe described here.

The hawki_detector_noise recipe takes one pair of raw DARK frames and one pair of raw FLAT_TWILIGHT frames as mandatory input. To get sensible results, each of these files must have identical DIT values and they all must have NDIT = 1. A MASTER_BPM or MASTER_CONF may be provided to ensure that bad pixels are not included in the recipe processing. The recipe compares the dark-corrected raw flat frames and calculates the readnoise, gain, and covariance (inter-pixel correlation) of each chip. Users are advised to select two flat fields of similar illumination that are bright enough to have a very high S/N ratio, but not so bright that they are near saturation. It is also recommended to run this recipe using data close in time to the science frames, because the detector readnoise and gain may vary over time.

For example, in order to create a MASTER_READGAIN file from the demo data, the SoF file could contain these files:

```
1> cat readgain.sof
$RAW_DATA_DIR/HAWKI.2008-11-18T09:23:08.501.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-18T09:23:20.840.fits DARK
$RAW_DATA_DIR/HAWKI.2008-11-17T23:44:47.342.fits FLAT_TWILIGHT
$RAW_DATA_DIR/HAWKI.2008-11-17T23:45:07.226.fits FLAT_TWILIGHT
$MASTER_CALIB_DATA_DIR/twilightconf_readgain.fits MASTER_CONF
```

where the environmental variables ('$' prefix) refer to the pathname of the user's data. Note that the MASTER_CONF frame here may not necessarily be the one described in §6.4.2 above. The FLAT_TWILIGHT files can be part of the set used to create a MASTER_CONF file to be used in this recipe, but this is not necessary. Another option is to use a MASTER_BPM which, in principle, should be the same for all filters. The recipe uses either of these two files only to identify and discard bad pixels.

The command to create the master readgain file using the default recipe input parameters is

```
1> esorex hawki_detector_noise readgain.sof
```

This will create one file:

readgain.fits: the MASTER_READGAIN pipeline product file.

and potentially an esorex.log file, depending on the user's esorex settings. A screenshot of the contents of the MASTER_READGAIN table is shown in Figure 6.10. Note that the 'GAIN' column in the table has **not** been corrected for covariance, i.e. the numbers in that column should be divided by those in the 'COVAR' column to obtain the actual value of the gain (see §9.5).

6.5 Standard Star Observations

This section describes how to process a set of HAWK-I images taken toward standard star fields with the hawki_standard_process recipe. This step is **not** required in order to process science observations;

Select All Invert	EXTNAME 10A Modify	READNOISE 1E ADUs Modify	GAIN 1E e-/ADU Modify	COVAR 1E Modify
1	CHIP1.INT1	9.846962E+00	2.101285E+00	1.306048E+00
2	CHIP2.INT1	8.990897E+00	2.174042E+00	1.240702E+00
3	CHIP3.INT1	8.409996E+00	2.047953E+00	1.246402E+00
4	CHIP4.INT1	8.549876E+00	2.406481E+00	1.209231E+00

Figure 6.10: Example HAWK-I table of MASTER_READGAIN values.

by default, science observations are calibrated photometrically using stars in the science fields themselves along with external catalogues (e.g. 2MASS). The recipe is provided to users who wish to use an alternative scheme for photometric calibration, e.g. for narrow-band filters.

The recipe takes a set of four STD observations as input. An additional seven calibration files are required as illustrated in Figure 6.2. Note that the PHOTCAL_TAB file must have data to calculate the conversion between the HAWK-I photometric system and the system used for photometric calibration, e.g. 2MASS. Astrometric and photometric calibration can be done either by using remote catalogues through CDS or by using a catalogue on a local machine. To use CDS, the recipe parameters cdssearch_astrom and/or cdsssearch_photom must specify a catalogue name, e.g. ppmxl. The astrometric catalogue is allowed to be different from the photometric catalogue. If a static calibration index file (e.g. MASTER_2MASS_CATALOGUE_ASTROM) is specified in the SoF **and** the cdssearch_astrom or cdssearch_photom parameters are not set to none, the recipe will use CDS and will ignore the index file. For moderately dense fields, the 2MASS catalogue is recommended for astrometry. In sparse fields, the WISE catalogue is recommended. For photometry, 2MASS is the recommended catalogue.

Users are **strongly cautioned** that, by default, all objects in the reference catalogue (e.g. 2MASS) are used for photometric calibration. If the reference catalogue is from a magnitude-limited survey that is shallower than the HAWK-I data, a bias will adversely affect this calibration at the ≈ 0.05 mag level. This bias is due to the large number of faint objects with large errors and/or inaccurate colours. This effect can be mitigated by excluding stars in the reference catalogue that have errors greater than a certain value. This threshold in error is specified by the magerrcut recipe parameter. A better approach may be to create a custom photometric reference catalogue by applying preferred colour and/or error cuts to a master catalogue, e.g. 2MASS.

The raw data are corrected for instrumental signatures, e.g. dark current and flat-fielding. The sky emission is removed using the "pawsky_mask" algorithm (see §9). The astrometry and photometry of individual images are calibrated with the user-specified catalogues. The calibrated, processed images (and corresponding variance maps) are created as output with the tags BASIC_CALIBRATED_STD and BASIC_VAR_MAP, respectively. For each processed image, a catalogue of detected objects is created with the tag BASIC_CAT_STD. One mean sky image and one corresponding variance map are created with the tags MEAN_SKY and MEAN_SKY_VAR, respectively.

Two types of standard star observations exist in HAWK-I data and both are supported by the HAWK-I pipeline. From the beginning of HAWK-I operations, the standard star template observed the same (single) standard star, placed in the centre of each detector, in a series of four exposures. From about mid-2015, the standard star template observes one of several 2MASS touchstone fields. These fields have been chosen to fill the HAWK-I

field of view with a large number of standard stars. However, for both templates there is no dithering or jittered images. As a result, the STD images are not combined in any way and therefore some artefacts may be present in the processed images.

For each processed image, there is an option to create a table of objects from the external catalogues that were used to calibrate the astrometry and photometry. These 'matched standards' tables have tags MATCHSTD_ASTROM and MATCHSTD_PHOTOM, respectively. The tables may be used to interpret the quality of the calibration and include columns of the difference between catalogued and image-derived positions and magnitudes.

An example of how to process the demo data for the standard star field 'FS32' is described below. It is assumed that the appropriate MASTER_DARK, MASTER_TWILIGHT_FLAT, MASTER_CONF, and MASTER_READGAIN have already been created. The SoF file might look like:

```
1> cat std.sof
$RAW_DATA_DIR/HAWKI.2008-11-18T00:12:19.100.fits STD
$RAW_DATA_DIR/HAWKI.2008-11-18T00:12:55.051.fits STD
$RAW_DATA_DIR/HAWKI.2008-11-18T00:13:31.577.fits STD
$RAW_DATA_DIR/HAWKI.2008-11-18T00:14:08.105.fits STD
$STD_CALIB_DIR/darkcomb.fits MASTER_DARK
$STD_CALIB_DIR/twilightconf.fits MASTER_CONF
$STD_CALIB_DIR/readgain.fits MASTER_READGAIN
$STD_CALIB_DIR/twilightcomb.fits MASTER_TWILIGHT_FLAT
$PPMXL_CAT_DIR/index_casu_ppmxl_astrom.fits MASTER_PPMXL_CATALOGUE_ASTROM
$TWOMASS_CAT_DIR/index_casu_2mass_photom.fits MASTER_2MASS_CATALOGUE_PHOTOM
$STATIC_CALIB_DATA_DIR/hawki_phot.fits PHOTCAL_TAB
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_ngp.fits SCHLEGEL_MAP_NORTH
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_sgp.fits SCHLEGEL_MAP_SOUTH
```

where the environmental variables ('$' prefix) refer to the pathname of the user's data.

For illustration purposes, the recipe parameter savemstd is changed from its default value (false) to true. The other parameters have their default value and are specified in a recipe configuration file:

```
1> cat hawki_standard_process.rc
hawki.hawki_standard_process.savemstd=true
hawki.hawki_standard_process.cdssearch_astrom=none
hawki.hawki_standard_process.cdssearch_photom=none
hawki.hawki_standard_process.minphotom=1
hawki.hawki_standard_process.prettynames=false
hawki.hawki_standard_process.preview_only=false
hawki.hawki_standard_process.psm_ipix=10
hawki.hawki_standard_process.psm_nbsize=256
hawki.hawki_standard_process.psm_niter=5
hawki.hawki_standard_process.psm_smkern=2.0
hawki.hawki_standard_process.psm_thresh=1.5
hawki.hawki_standard_process.src_cat_icrowd=TRUE
```

```
hawki.hawki_standard_process.src_cat_ipix=10
hawki.hawki_standard_process.src_cat_nbsize=128
hawki.hawki_standard_process.src_cat_rcore=10.0
hawki.hawki_standard_process.src_cat_thresh=1.5
```

The command to create a calibrated image using these parameters is

```
1> esorex --recipe-config=hawki_standard_process.rc hawki_standard_process
   std.sof
```

This will create 22 files:

`exp_[1-4].fits`: the four `BASIC_CALIBRATED_STD` product files; one for each input `STD` file. The number in the file name reflects to the order of appearance of the files in the SoF.

`exp_var_[1-4].fits`: the four `BASIC_VAR_MAP` product files corresponding to each `BASIC_CALIBRATED_STD` file.

`exp_cat[1-4].fits`: the four `BASIC_CAT_STD` product files corresponding to each `BASIC_CALIBRATED_STD` file.

`sky_1.fits`: the `MEAN_SKY` product file.

`sky_var_1.fits`: the `MEAN_SKY_VAR` product file.

`mstd_a[1-4].fits`: the four `MATCHSTD_ASTROM` files corresponding to each `BASIC_CALIBRATED_STD` file.

`mstd_p[1-4].fits`: the four `MATCHSTD_PHOTOM` files corresponding to each `BASIC_CALIBRATED_STD` file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true'. A screenshot of the first chip from one of the processed images is shown in Figure 6.11.

6.6 Science Observations

There are two recipes for processing HAWK-I science images: `hawki_science_process` and `hawki_science_postprocess`. A typical sequence of HAWK-I science images consists of two or more observations that are offset from each other by a small angular distance ('jitter'); the magnitude of each jitter is usually a random number bounded by some user-defined limit. The `hawki_science_process` recipe combines a sequence of raw, jittered images into one 'stacked' image. This combination is done separately for each chip, i.e. a stack comprises four FITS extensions (one for each chip). The recipe also creates one fully processed and calibrated image for each raw input image.

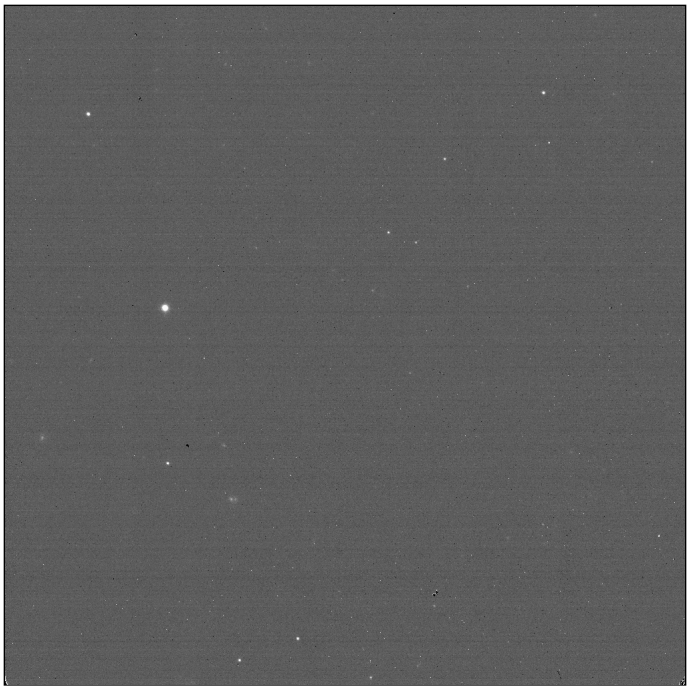

Figure 6.11: Processed STD image; the first chip of the first STD image in the example SoF is shown.

By contrast, the hawki_science_postprocess recipe combines a sequence of *calibrated*, jittered images into one 'tile'. This includes all chips together into the PHU, i.e. a tile has no FITS extensions. The recipe may also correct for varying background emission when creating a catalogue of objects in the tile ('nebulise'). If a user does not want to create a tile, the hawki_science_postprocess recipe does not need to be run; a stacked image is fully calibrated. However, if a user does want to create a tile or a catalogue of 'nebulised' objects, both recipes must be run in series.

Users are **strongly cautioned** that, by default, all objects in the reference catalogue (e.g. 2MASS) are used for photometric calibration. If the reference catalogue is from a magnitude-limited survey that is shallower than the HAWK-I data, a bias will adversely affect this calibration at the ≈ 0.05 mag level. This bias is due to the large number of faint objects with large errors and/or inaccurate colours. This effect can be mitigated by excluding stars in the reference catalogue that have errors greater than a certain value. This threshold in error is specified by the magerrcut recipe parameter. A better approach may be to create a custom photometric reference catalogue by applying preferred colour and/or error cuts to a master catalogue, e.g. 2MASS.

6.6.1 Creating a stacked image with hawki_science_process

Each of the raw, jittered (OBJECT) images are first corrected for instrumental signatures, e.g. dark current and flat-fielding. The sky emission is removed using offset sky observations and/or a user-specified algorithm (see §9). The astrometry and photometry of individual images are calibrated with the user-specified catalogues. For each raw input image, a calibrated image (and corresponding variance map) is created with the tags

BASIC_CALIBRATED_SCI and BASIC_VAR_MAP, respectively. For each of these calibrated images, there is an option for a catalogue of detected objects to be created with the tag OBJECT_CATALOGUE_SCI. One or more mean sky images and corresponding variance maps are created with the tags MEAN_SKY and MEAN_SKY_VAR, respectively. The number of mean sky images is determined by the total elapsed time over which the sequence of input OBJECT files were acquired; one mean sky image and one variance map are created from at least 8 images and every $\approx$ 30 minutes of elapsed time.

If a user supplies the recipe with a set of offset sky observations, the recipe will create calibrated versions of each of those input sky frames (and variance maps) with tags BASIC_CALIBRATED_SKY and BASIC_VAR_MAP_SKY, respectively. In addition, the sky emission in the science images will be removed using **only** the offset skies. The stacked image and associated variance and confidence maps are created with tags JITTERED_IMAGE_SCI, JITTERED_VAR_IMAGE, and CONFIDENCE_MAP_JITTERED, respectively. A catalogue of objects detected in the stack is created with the tag OBJECT_CATALOGUE_JITTERED.

For each processed image, including the stack, there is an option to create a table of objects from the external catalogues that were used to calibrate the astrometry for those images. The astrometric 'matched standard' tables have the tag MATCHSTD_ASTROM. The tables may be used to assess the quality of the calibration and include columns with the angular offset between the catalogued and derived positions.

If requested, a matched standard table containing the photometric calibrators may created for the stack as well; this file has the tag MATCHSTD_PHOTOM. The table can be used to assess the quality of the calibration and can also be used to refine the photometric calibration of the stack. If the recipe is run with this matched standard table as an input, the recipe will only use objects in that table (with photometric errors less than magerrcut) to calibrate the photometry of objects in the stack. For example, if a stellar artefact or a very bright star with large photometric error in 2MASS was used in the calibration, those objects may removed from the table before re-running the recipe with the updated matched standard table as input. The removal of objects (rows) from the table should be done with any FITS-compliant editor, e.g. fv[3] or TOPCAT[4].

The recipe takes a set of OBJECT files observations as mandatory input. A set of optional offset SKY files may be also be provided. An additional seven calibration files are required as illustrated in Figure 6.3. Note that the PHOTCAL_TAB file must have data to calculate the conversion between the HAWK-I photometric system and the system used for photometric calibration, e.g. 2MASS. Astrometric and photometric calibration can be done either by using remote catalogues through CDS or by using a catalogue on a local machine. To use CDS, the recipe parameters cdssearch_astrom and/or cdsssearch_photom must specify a catalogue name, e.g. ppmxl. The astrometric catalogue is allowed to be different from the photometric catalogue. If a static calibration index file (e.g. MASTER_2MASS_CATALOGUE_ASTROM) is specified in the SoF **and** the cdssearch_astrom or cdssearch_photom parameters are not set to none, the recipe will use CDS and will ignore the index file. For moderately dense fields, the 2MASS catalogue is recommended for astrometry. In sparse fields, the WISE catalogue is recommended. For photometry, 2MASS is the recommended catalogue.

If a user wishes to estimate the sky using the pawsky_mask_pre algorithm, a MASTER_OBJMASK file may be specified in the SoF (see §9.7).

If a user would like to skip the sky creation performed by the recipe, a processed sky image and variance map may be specified with tags MASTER_SKY and MASTER_SKY_VAR, respectively. To ensure an accurate sky

[3] http://heasarc.gsfc.nasa.gov/ftools/fv/
[4] http://www.star.bris.ac.uk/~mbt/topcat/

subtraction, it is recommended that these master sky files come from the recipe itself (but is not necessary). Note that in this instance, the sky product files (e.g., `MEAN_SKY`) have a different tag than the input files (e.g. `MASTER_SKY`). Skipping the sky creation in the recipe may result in a significantly faster recipe execution time.

Before running this recipe, users are strongly advised to first run it with `preview_only = true` to see how the recipe will process the data, create skies, etc.

An example of how to process the demo data for the target 'BDF4' is described below. It is assumed that the appropriate `MASTER_DARK, MASTER_TWILIGHT_FLAT, MASTER_CONF` and `MASTER_READGAIN` have already been created. The SoF file might look like:

```
1> cat stack.sof
$RAW_DATA_DIR/HAWKI.2008-11-18T00:22:15.321.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:25:09.307.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:28:04.630.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:30:58.617.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:33:52.594.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:36:46.623.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:39:40.641.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:42:34.669.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:45:29.959.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:48:25.298.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:51:17.971.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:54:13.268.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T00:57:07.244.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:00:01.276.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:02:55.252.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:05:49.285.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:08:43.304.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:11:37.281.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:14:32.605.fits OBJECT
$RAW_DATA_DIR/HAWKI.2008-11-18T01:17:26.639.fits OBJECT
$SCI_CALIB_DIR/darkcomb.fits MASTER_DARK
$SCI_CALIB_DIR/twilightconf.fits MASTER_CONF
$SCI_CALIB_DIR/readgain.fits MASTER_READGAIN
$SCI_CALIB_DIR/twilightcomb.fits MASTER_TWILIGHT_FLAT
$PPMXL_CAT_DIR/index_casu_ppmxl_astrom.fits MASTER_PPMXL_CATALOGUE_ASTROM
$TWOMASS_CAT_DIR/index_casu_2mass_photom.fits MASTER_2MASS_CATALOGUE_PHOTOM
$STATIC_CALIB_DATA_DIR/hawki_phot.fits PHOTCAL_TAB
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_ngp.fits SCHLEGEL_MAP_NORTH
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_sgp.fits SCHLEGEL_MAP_SOUTH
```

where the environmental variables (`'$'` prefix) refer to the pathname of the user's data.

For illustration purposes, the recipe parameters `savecat` and `savemstd` are changed from their default value ('false') to 'true'. The other parameters are set to their default values and are specified in a recipe configuration file:

```
1> cat hawki_science_process.rc
hawki.hawki_science_process.savecat=true
hawki.hawki_science_process.savemstd=true
hawki.hawki_science_process.cdssearch_astrom=none
hawki.hawki_science_process.cdssearch_photom=none
hawki.hawki_science_process.minphotom=1
hawki.hawki_science_process.skyalgo=auto
hawki.hawki_science_process.psm_ipix=10
hawki.hawki_science_process.psm_nbsize=128
hawki.hawki_science_process.psm_niter=5
hawki.hawki_science_process.psm_smkern=2.0
hawki.hawki_science_process.psm_thresh=1.5
hawki.hawki_science_process.stk_cat_icrowd=true
hawki.hawki_science_process.stk_cat_ipix=10
hawki.hawki_science_process.stk_cat_nbsize=128
hawki.hawki_science_process.stk_cat_rcore=10.0
hawki.hawki_science_process.stk_cat_thresh=1.5
hawki.hawki_science_process.stk_fast=auto
hawki.hawki_science_process.stk_hthr=5.0
hawki.hawki_science_process.stk_lthr=5.0
hawki.hawki_science_process.stk_method=linear
hawki.hawki_science_process.stk_seeing=false
hawki.hawki_science_process.prettynames=false
hawki.hawki_science_process.preview_only=false
```

The command to create the pipeline products with these parameters is

```
1> esorex --recipe-config=hawki_science_process.rc hawki_science_process
   stack.sof
```

This will create 90 files:

`exp_[1-20].fits`: the twenty `BASIC_CALIBRATED_SCI` product files; one for each input `OBJECT` file. The number in the filename reflects the chronological order of the files in the SoF.

`exp_var_[1-20].fits`: the twenty `BASIC_VAR_MAP` product files associated with the `BASIC_CALIBRATED_SCI` files.

`exp_cat[1-20].fits`: the twenty `OBJECT_CATALOGUE_SCI` product files associated with the `BASIC_CALIBRATED_SCI` files

`exp_mstd_a[1-20].fits`: the twenty `MATCHSTD_ASTROM` files associated with the `BASIC_CALIBRATED_SCI` files.

`sky_[1-2].fits`: the two `MEAN_SKY` product files.

`sky_var_[1-2].fits`: the two `MEAN_SKY_VAR` product files.

`stack_1.fits`: the `JITTERED_IMAGE_SCI` product file.

`stack_conf_1.fits`: the `CONFIDENCE_MAP_JITTERED` product file.

`stack_var_1.fits`: the `JITTERED_VAR_IMAGE` product file.

`stack_cat_1.fits`: the `OBJECT_CATALOGUE_JITTERED` product file.

`stack_mstd_a1.fits`: the `MATCHSTD_ASTROM` file associated with the `JITTERED_IMAGE_SCI` file.

`stack_mstd_p1.fits`: the `MATCHSTD_PHOTOM` file associated with the `JITTERED_IMAGE_SCI` file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true'. A screenshot of the stacked image is shown in Figure 6.12.

6.6.2 Creating a tiled image with hawki_science_postprocess

A tile is created by combining a set of calibrated, jittered images (tags `BASIC_CALIBRATED_SCI` and `BASIC_VAR_MAP`). After the images are combined, the resulting tile is calibrated astrometrically and photometrically with user-specified catalogues. The tile and corresponding variance and confidence maps are created with the tags `TILED_IMAGE`, `TILED_VAR_MAP`, and `TILED_CONFIDENCE_MAP`, respectively. A catalogue of objects detected in the tile has the tag `TILED_OBJECT_CATALOGUE`.

The tile may show emission in the background that varies on different spatial scales. This may be due to incomplete sky subtraction and/or diffuse astrophysical nebular emission. It may affect the accuracy of photometric measurements of objects in the image, especially in regions where the background varies strongly on small scales. There is a recipe parameter that offers the option of removing the effect of this variable background before creating the object catalogue through a process called 'nebulising' (see §9). If this option is used, only the object catalogue is affected; the output tiled image retains the variable background emission.

There is also an option to create a table of objects from the external catalogues that were used to calibrate the astrometry and photometry of the tile (the `savemstd` recipe parameter). The astrometric 'matched standard' table has the tag `MATCHSTD_ASTROM`. The table may be used to assess the quality of the calibration. The matched standard table with the photometric calibrators can be used to refine the photometric calibration of the tile; this file has the tag `MATCHSTD_PHOTOM`. If the recipe is run with this table as an input, the recipe will only use objects in that table (with photometric errors less than `magerrcut`) to calibrate the photometry of objects in the tile. For example, if a stellar artefact or a very bright star with large photometric error in 2MASS was used in the calibration, those objects may removed from the table before re-running the recipe with the updated matched standard table as input. The removal of objects (rows) from the table should be done with any FITS-compliant editor, e.g. fv or TOPCAT.

The recipe takes a set of `BASIC_CALIBRATED_SCI` and `BASIC_VAR_MAP` files as mandatory input. An additional four calibration files are required as illustrated in Figure 6.3. Note that the `PHOTCAL_TAB` file must have data to calculate the conversion between the HAWK-I photometric system and the system used for photometric calibration, e.g. 2MASS. Astrometric and photometric calibration can be done either by using

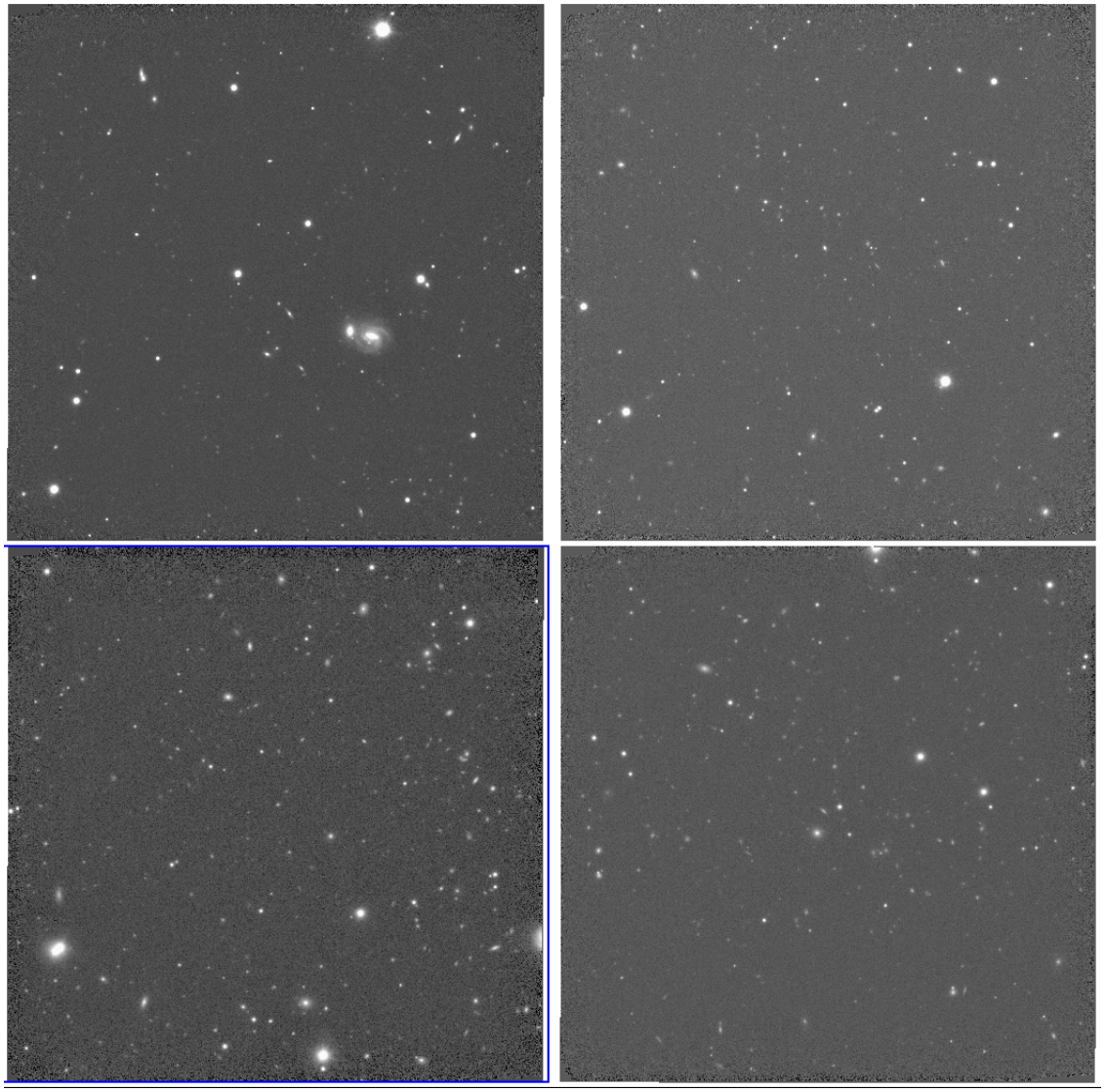

Figure 6.12: A stack created by `hawki_science_process` from twenty `OBJECT` images in the demo dataset. All four chips are shown.

remote catalogues through CDS or by using a catalogue on a local machine. To use CDS, the recipe parameters `cdssearch_astrom` and/or `cdsssearch_photom` must specify a catalogue name, e.g. `ppmxl`. The astrometric catalogue is allowed to be different from the photometric catalogue. If a static calibration index file (e.g. `MASTER_2MASS_CATALOGUE_ASTROM`) is specified in the SoF **and** the `cdssearch_astrom` or `cdssearch_photom` parameters are not set to `none`, the recipe will use CDS and will ignore the index file. For moderately dense fields, the 2MASS catalogue is recommended for astrometry. In sparse fields, the WISE catalogue is recommended. For photometry, 2MASS is the recommended catalogue.

An example of how to create a tile from the demo data toward the target 'BDF4' is described below; this uses the output of the recipe described in §6.6.1 above. It is assumed that the appropriate `MASTER_CONF` has already been created. The SoF file might look like:

```
1> cat tile.sof
$SCI_PRO_DIR/exp_1.fits BASIC_CALIBRATED_SCI
$SCI_PRO_DIR/exp_2.fits BASIC_CALIBRATED_SCI
...
...
$SCI_PRO_DIR/exp_20.fits BASIC_CALIBRATED_SCI
$SCI_PRO_DIR/exp_var_1.fits BASIC_VAR_MAP
$SCI_PRO_DIR/exp_var_2.fits BASIC_VAR_MAP
...
...
$SCI_PRO_DIR/exp_var_20.fits BASIC_VAR_MAP
$SCI_CALIB_DIR/twilightconf.fits MASTER_CONF
$PPMXL_CAT_DIR/index_casu_ppmxl_astrom.fits MASTER_PPMXL_CATALOGUE_ASTROM
$TWOMASS_CAT_DIR/index_casu_2mass_photom.fits MASTER_2MASS_CATALOGUE_PHOTOM
$STATIC_CALIB_DATA_DIR/hawki_phot.fits PHOTCAL_TAB
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_ngp.fits SCHLEGEL_MAP_NORTH
$STATIC_CALIB_DATA_DIR/SFD_dust_4096_sgp.fits SCHLEGEL_MAP_SOUTH
```

where the environmental variables (`'$'` prefix) refer to the pathname of the user's data. For brevity, ellipses are used in place of the full range of `BASIC_CALIBRATED_SCI` and `BASIC_VAR_MAP` files.

For illustration purposes, the recipe parameter `savemstd` is changed from its default value ('false') to 'true'. The other parameters are set to their default values and are specified in a recipe configuration file:

```
1> cat hawki_science_postprocess.rc
hawki.hawki_science_postprocess.savemstd=true
hawki.hawki_science_postprocess.cat_icrowd=true
hawki.hawki_science_postprocess.cat_ipix=4
hawki.hawki_science_postprocess.cat_nbsize=64
hawki.hawki_science_postprocess.cat_rcore=3.0
hawki.hawki_science_postprocess.cat_thresh=1.25
hawki.hawki_science_postprocess.cdssearch_astrom=none
hawki.hawki_science_postprocess.cdssearch_photom=none
hawki.hawki_science_postprocess.minphotom=20
```

```
hawki.hawki_science_postprocess.neb_linfilt=33
hawki.hawki_science_postprocess.neb_medfilt=101
hawki.hawki_science_postprocess.nebulise=false
hawki.hawki_science_postprocess.prettynames=false
hawki.hawki_science_postprocess.stk_fast=auto
```

The command to create the pipeline products using these parameters is

```
1> esorex --recipe-config=hawki_science_postprocess.rc
   hawki_science_postprocess tile.sof
```

This will create 6 files:

`tile_1.fits`: the `TILED_IMAGE` product file.

`tile_cat1.fits`: the `TILED_OBJECT_CATALOGUE` product file.

`tile_conf_1.fits`: the `TILED_CONFIDENCE_MAP` product file.

`tile_mstd_a1.fits`: the `MATCHSTD_ASTROM` product file.

`tile_mstd_p1.fits`: the `MATCHSTD_PHOTOM` product file.

`tile_var1.fits`: the `TILED_VAR_MAP` product file.

and potentially an `esorex.log` file, depending on the user's `esorex` settings. Note that the filenames will be different from those above if the parameter `prettynames` is set to 'true'. A screenshot of the tile is shown in Figure 6.13.

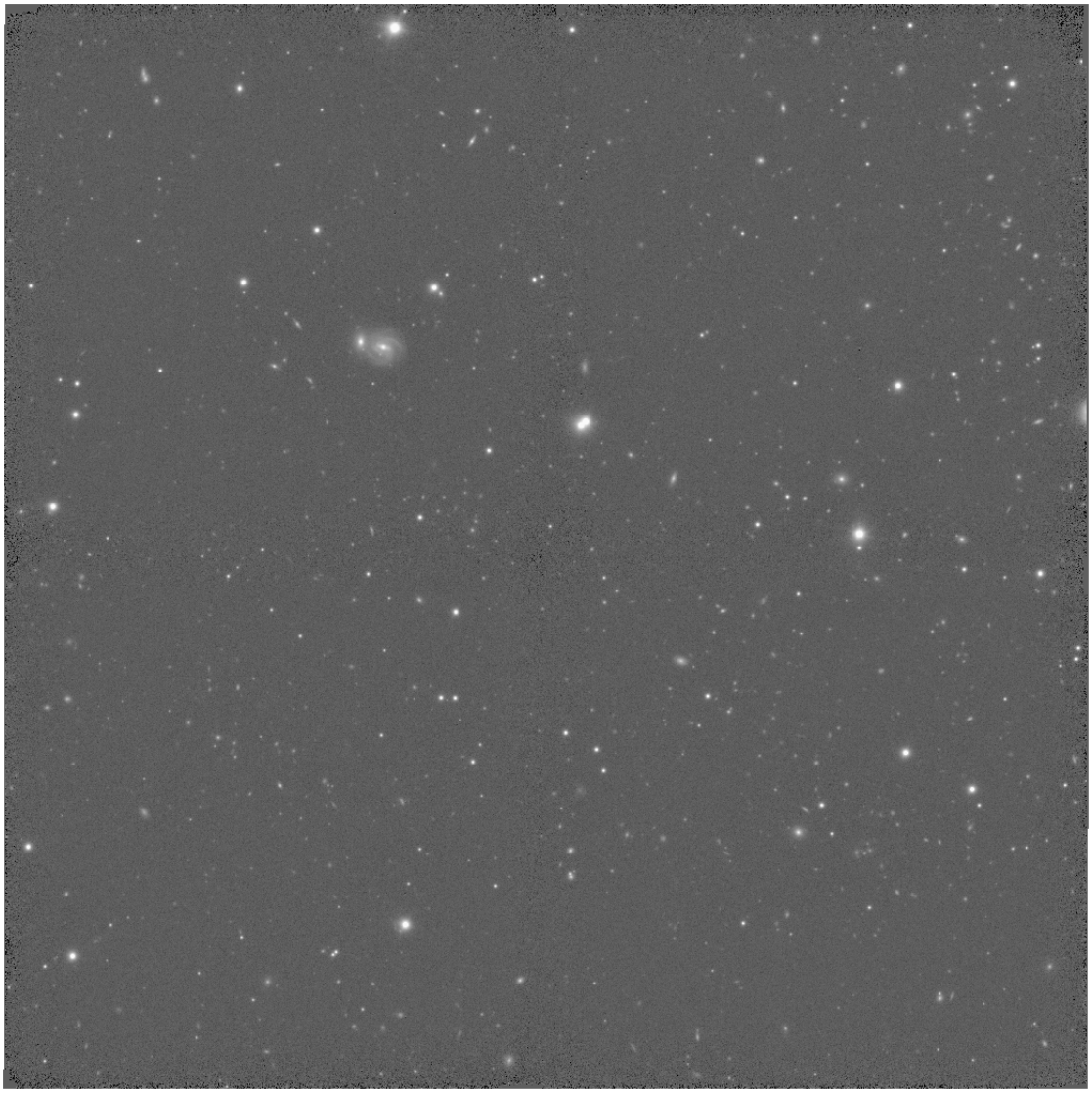

Figure 6.13: A tile created by `hawki_science_postprocess` toward the target 'BDF4' in the demo dataset.

7 Troubleshooting

This section describes the cause of, and solution to, some common error messages.

7.1 Log file

All recipes run through `esorex` will generate a logfile with varying levels of information about what the recipe is doing, warnings, etc. The messages that appear in the logfile are prefixed by a timestamp and one of the following:

`[ INFO ]` Informational message about what the recipe is doing, e.g. creating output files.

`[ DEBUG ]` More detailed message about what the recipe is doing, e.g. a statistical description of an intermediate data product. These kind of messages will only appear if a user changes the `esorex` default settings for `msg-level` and/or `log-level` to `debug`. When submitting bug reports, it is recommended that the `debug` option be turned on.

`[WARNING]` A non-fatal error message that indicates something may have gone wrong during the processing, e.g. not enough photometric standards were found during calibration. A user is advised to check the data products carefully if a `WARNING` is issued.

`[ ERROR ]` A fatal error that indicates a serious problem occurred during the processing and the recipe has stopped, e.g. the recipe configuration file cannot be parsed.

The logfile should be submitted with any bug report.

7.2 Sky subtraction

If only one `OBJECT` or `STD` file is provided to a recipe, and a `MASTER_SKY` is not provided, no sky subtraction is possible. A user must provide more than one science/standard frame or a master sky frame in order to estimate the sky emission.

If the `OBJECT` frames are not jittered, a sky estimate will be created. However, the sky will not be accurate and the resultant calibrated images will not be terribly useful. The solution to this is to either provide a `MASTER_SKY` frame or to skip the sky estimation altogether (recipe parameter `skyalgo = none`).

7.3 Astrometry in crowded fields

In crowded fields, a model fit to the WCS can fail or have a large error. If a user is using the WISE catalogue for astrometry, we recommend a less dense reference catalogue, e.g. 2MASS. It may also help to use the recipe parameter `stk_cat_icrowd = true`.

8 Recipe Reference

This section provides a detailed description of each HAWK-I recipe, including input files, parameters, output files, and QC parameters. A description of some of the common algorithms used by one or more recipes can be found in §9. The file format specifications are provided in §A. Note that the parameter type "float" below refers to a 32-bit floating point number.

8.1 Common recipe QC parameters

All recipes contain the following QC parameters in their products:

`QC.DET.ID`: Modification of the `EXTNAME` header keyword, where the part after `CHIP` is dropped; e.g., `CHIP1` instead of `CHIP1.INT1`

`QC.PROC.SCHEME`: 'CASU_pipeline' for all recipes except hawki_science_process and hawki_science_postprocess, where it is 'HAWKI_R'

`QC.NDSAMPLES`: If the keyword `ESO DET NCORRS NAME` in the PHU has the value 'DoubleRdRstRd', then this is set to 0. Otherwise, it is set to the value of the keyword `ESO DET NDSAMPLES` from the PHU.

8.2 hawki_dark_combine

8.2.1 Description

Combine a list of dark frames into a mean dark frame. Optionally compare the output frame to a reference dark frame. A description of some of the common algorithms used by this recipe can be found in §9.

8.2.2 Input frames

Frame tag	Constraint	# files
DARK	Required	> 1
REFERENCE_DARK	Optional	1
MASTER_BPM	Optional	1
MASTER_CONF	Optional	1

Note that if both `MASTER_BPM` and `MASTER_CONF` are given, the `MASTER_CONF` will be used. Also note that in practice, the number of raw `DARK` files should be at least 5 in order to get useful results.

8.2.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
`combtype`	string	**median**, mean	When combining images, use the mean or median pixel value.
`scaletype`	string	**additive**, none	When combining images, scale each one by parameter value to bring background values into agreement
`xrej`	bool	**true**, false	If true, an extra rejection cycle is performed after combination, i.e. an extra pass ensures that what has been clipped deserves to have been clipped.
`thresh`	float	**5.0**, $0 < x$	During combination, reject pixels more than `thresh` sigma above the background
`ncells`	int	**8**, 1,2,4,16,32,64	If a reference dark is provided, divide each 128pix channel on each chip into `ncells` when calculating the statistical difference between the reference dark and master dark.
`prettynames`	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.

8.2.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
MASTER_DARK	darkcomb.fits	dark_<NIGHT>_<DIT>_<NDIT>.fits	Master dark image
DIFFIMG_DARK	darkdiff.fits	darkdiff_<NIGHT>_<DIT>_<NDIT>.fits	If a reference dark is provided in SoF, this is the difference between the master dark and reference dark, i.e. (MASTER - REFERENCE).
DIFFIMG_STATS_DARK	darkdifftab.fits	darkdifftab_<NIGHT>_<DIT>_<NDIT>.fits	If a reference dark is provided in SoF, this is a table with statistical description of the difference image within small areas of each chip.

Note that <NIGHT> refers to the eight digit representation of the 'civil' night (local time starting at midday) on which the data were taken, e.g. 20140820.

8.2.5 Quality control parameters

The following QC parameters are found in each extension of the `MASTER_DARK`:

`QC.RON?`: Readnoise estimate from pair. There is one of these keywords for each input DARK

`QC.RAW MEAN MEAN`: Mean of raw image means

`QC.RAW MEAN MEDIAN`: Mean of raw image medians

`QC.RAW RMS MEAN`: RMS of raw image means

`QC.DARK MED`: Median of master dark frame

`QC.DARK MEAN`: Mean of master dark frame

`QC.DARK STDEV`: RMS of mean dark frame

`QC.PARTICLE_RATE`: [N/(detector*sec)] where N is number of objects classified as cosmic rays

`QC.DARK NBADPIX`: Number of hot pixels, where a hot pixel is one that is consistently too high and is not corrected for in the flat field

`QC.BADFRAC`: Hot pixel fraction

The following QC parameters are found in each extension of `DIFFIMG_DARK`:

`QC.RON?`: Readnoise estimate from pair. There is one of these keywords for each input DARK

`QC.RAW MEAN MEAN`: Mean of raw image means

`QC.RAW MEAN MEDIAN`: Mean of raw image medians

`QC.RAW RMS MEAN`: RMS of raw image means

`QC.DARKDIFF_MED`: Median of difference image

`QC.DARKDIFF_RMS`: RMS of difference image

The following QC parameters are found in the last three extensions of `DIFFIMG_STATS_DARK`:

`QC.RON?`: Readnoise estimate from pair. There is one of these keywords for each input DARK

`QC.RAW MEAN MEAN`: Mean of raw image means

`QC.RAW MEAN MEDIAN`: Mean of raw image medians

`QC.RAW RMS MEAN`: RMS of raw image means

The mean and RMS value of several of these keywords are stored in the PHU of the MASTER_DARK product. The average of the keyword value QC.KW across all detectors/extensions is stored as QC.KW.AVG; the RMS value is stored as QC.KW.RMS. The following AVG/RMS values are computed and stored in the PHU:

QC.RAW.MEAN.MEAN.AVG/RMS

QC.RAW.MEAN.MEDIAN.AVG/RMS

QC.RAW.RMS.MEAN.AVG/RMS

QC.DARK.NBADPIX.AVG/RMS

QC.DARK.MED.AVG/RMS

QC.DARK.MEAN.AVG/RMS

QC.DARK.STDEV.AVG/RMS

QC.RON1.AVG/RMS

QC.RON2.AVG/RMS

QC.PARTICLE.RATE.AVG/RMS

QC.BADFRAC.AVG/RMS

8.3 hawki_twilight_flat_combine

8.3.1 Description

Combine a list of twilight flat frames into a mean frame. Optionally compare the output frame to a master twilight flat frame. A description of some of the common algorithms used by this recipe can be found in §9.

8.3.2 Input frames

Frame tag	Constraint	# files
FLAT_TWILIGHT	Required	> 1
MASTER_DARK	Required	1
REFERENCE_TWILIGHT_FLAT	Optional	1
MASTER_BPM	Optional	1
MASTER_CONF	Optional	1

For best results, it is strongly recommended that at least 10 raw FLAT_TWILIGHT files be included in a SoF, each with counts between 10,000-20,000 ADUs. If both MASTER_BPM and MASTER_CONF are provided, only the MASTER_CONF will be used.

8.3.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
`combtype`	string	**median**, mean	When combining images, use the mean or median pixel value.
`scaletype`	string	**multiplicative**, additive, exptime, none	When combining images, scale each one by parameter value to bring background values into agreement
`lthr`	float	**4000**, any float	Discard pixels with values below `lthr` ADUs in the raw flat images.
`hthr`	float	**30000**, any float	Discard pixels with values higher than `hthr` ADUs in the raw flat images.
`thresh`	float	**5.0**, $0 < x$	During combination, reject pixels more than `thresh` sigma above the background
`xrej`	bool	**true**, false	If true, an extra rejection cycle is performed after combination, i.e. an extra pass ensures that what has been clipped deserves to have been clipped.
`ncells`	int	**8**, 1,2,4,16,32,64	If a reference dark is provided, divide each 128pix channel on each chip into `ncells` when calculating the statistical difference between the reference dark and master dark.
`prettynames`	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.

8.3.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
MASTER_TWILIGHT_FLAT	twilightcomb.fits	<FILT>_flat_<NIGHT>.fits	Combined twilight flat image
MASTER_CONF	twilightconf.fits	<FILT>_conf_<NIGHT>.fits	Confidence map derived from raw flat images
MASTER_BPM	bpmmap.fits	<FILT>_bpm_<NIGHT>.fits	Bad pixel map derived from raw flat images
RATIOIMG_TWILIGHT_FLAT	twilightratio.fits	<FILT>_ratio_<NIGHT>.fits	If a reference flat is provided in SoF, this is the ratio of the master dark to the reference flat, i.e. (MASTER/REFERENCE).
RATIOIMG_STATS_TWILIGHT_FLAT	twilightratiotab.fits	<FILT>_ratiotab_<NIGHT>.fits	If a reference flat is provided in SoF, this is a table with statistical description of the ratio image within small areas of each chip.

Note that <NIGHT> refers to the eight digit representation of the 'civil' night (local time starting at midday) on which the data were taken, e.g. 20140820.

8.3.5 Quality control parameters

The following QC parameters are found in the `MASTER_TWILIGHT_FLAT`:

`QC.FLAT NORM`: Median value before normalisation

`QC.FLATRMS`: RMS of output flat

`QC.FLAT MEDMIN`: Ensemble minimum

`QC.FLAT MEDMAX`: Ensemble maximum

`QC.FLAT MEDMEAN`: Ensemble average

`QC.FLAT MEDMED`: Ensemble median

`QC.FLAT MEDSTDEV`: Ensemble sigma

`QC.FLUX RATIO`: max/min ratio

`QC.FLATRNG`: Ensemble range

`QC.TWIPHOT`: [adu] Estimated photon noise

`QC.TWISNRATIO`: Estimated S/N ratio

The following QC parameters are found in each extension of the `RATIOIMG_TWILIGHT_FLAT`:

`QC.FLAT NORM`: Median value before normalisation

`QC.FLATRATIO_MED`: Median of ratio map

`QC.FLATRATIO_RMS`: RMS of ratio map

The following QC parameters are found in each extension of the `RATIOIMG_STATS_TWILIGHT_FLAT` and `MASTER_CONF`:

`QC.FLAT NORM`: Median value before normalisation

The following QC parameters are found in each extension of the `MASTER_BPM`:

`QC.FLAT NORM`: Median value before normalisation

`QC.FLAT NBADPIX`: Number of bad pixels detected

`QC.FLAT BADFRAC`: Fraction of bad pixels detected

The mean and RMS value of several of these keywords are stored in the PHU of the MASTER_DARK product. The average of the keyword value QC.KW across all detectors/extensions is stored as QC.KW.AVG; the RMS value is stored as QC.KW.RMS. The following AVG/RMS values are computed and stored in the PHU:

QC.FLAT.NORM.AVG/RMS

QC.FLAT.MEDMIN.AVG/RMS

QC.FLAT.MEDMAX.AVG/RMS

QC.FLAT.MEDMEAN.AVG/RMS

QC.FLAT.MEDMED.AVG/RMS

QC.FLAT.MEDSTDEV.AVG/RMS

QC.FLAT.NBADPIX.AVG/RMS

QC.FLATRNG.AVG/RMS

QC.TWIPHOT.AVG/RMS

QC.TWISNRATIO.AVG/RMS

8.4 hawki_detector_noise

8.4.1 Description

Compute the readnoise and gain for the HAWK-I detectors using two dark frames and two flat field frames. Both must have been exposed with DET.NDIT = 1. A description of some of the common algorithms used by this recipe can be found in §9.

8.4.2 Input frames

Frame tag	Constraint	# files
FLAT_TWILIGHT	Required	2
DARK	Required	2
MASTER_BPM	Optional	1
MASTER_CONF	Optional	1

If both MASTER_BPM and MASTER_CONF are provided, only the MASTER_CONF will be used. Users are advised to select two flat fields of similar illumination that are bright enough to have a very high S/N ratio, but not so bright that they are near saturation. It is also recommended to run this recipe using data close in time to the science frames, because the detector readnoise and gain may vary over time.

8.4.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
`thresh`	float	**5.0**, $0 < x$	During combination, reject pixels more than `thresh` sigma above the background

8.4.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
`MASTER_READGAIN`	`readgain.fits`	N/A	Table containing readnoise, gain, and covariance for each chip

8.4.5 Quality control parameters

This recipe does not create products with QC parameters.

8.5 hawki_standard_process

8.5.1 Description

Process a stack for HAWK-I standard data. Remove instrumental signature, remove sky background, photometrically and astrometrically calibrate each image in the stack individually. A description of some of the common algorithms used by this recipe can be found in §9.

8.5.2 Input frames

Frame tag	**Constraint**	**# files**
`STD`	Required	$1 \leq n \leq 4$
`MASTER_DARK`	Required	1
`MASTER_TWILIGHT_FLAT`	Required	1
`MASTER_CONF`	Required	1
`MASTER_READGAIN`	Required	1
`PHOTCAL_TAB`	Required	1
`SCHLEGEL_MAP_NORTH`	Required	1
`SCHLEGEL_MAP_SOUTH`	Required	1
`MASTER_2MASS_CATALOGUE_ASTROM`	Optional	1

continued on next page

Frame tag	Constraint	# files
MASTER_PPMXL_CATALOGUE_ASTROM	Optional	1
MASTER_LOCAL_CATALOGUE_ASTROM	Optional	1
MASTER_2MASS_CATALOGUE_PHOTOM	Optional	1
MASTER_PPMXL_CATALOGUE_PHOTOM	Optional	1
MASTER_LOCAL_CATALOGUE_PHOTOM	Optional	1

If both MASTER_2MASS_CATALOGUE_ASTROM and MASTER_PPMXL_CATALOGUE_ASTROM are in the SoF, only MASTER_2MASS_CATALOGUE_ASTROM will be used. Also note that providing only one STD file is allowed, but this will yield less than ideal pipeline products.

8.5.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
preview_only	bool	**false**, true	If true, don't run recipe but instead print out a summary of how data will be processed
minphotom	int	**1**, $1 \leq n \leq 10^5$	During photometric calibration, at least minphotom number of standards must be used.
prettynames	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.
savemstd	bool	**false**, true	If true, create matched standard catalogues
cdssearch_astrom	string	**none**, 2mass, usnob, ppmxl, wise	If not none, retrieve subset of specified catalogue from CDS for astrometric calibration
cdssearch_photom	string	**none**, 2mass, ppmxl	If not none, retrieve subset of specified catalogue from CDS for photometric calibration
cat_ipix	int	**10**, $1 \leq n \leq 10^5$	min. number of pixels for catalogued objects
cat_thresh	float	**1.5**, $0 < x$	detection threshold above sky in units of σ above bkg for catalogued objects. The threshold is applied to the flux within each aperture.

continued on next page

Parameter	Type	Values **default**,other	Description
`cat_icrowd`	bool	**true**, false	if true, use deblending for catalogued objects
`cat_rcore`	float	**10.0**, $0 < x \leq 1024$	size of standard aperture in units of pixels for catalogued objects
`cat_nbsize`	int	**128**, $1 \leq x \leq 2048$	size of smoothing box when estimating bkg for catalogued objects
`psm_ipix`	int	**10**, $1 \leq n \leq 10^5$	for sky estimation, min. number of pixels for objects
`psm_niter`	int	**5**, $1 \leq n \leq 10$	for sky estimation, number of iterations
`psm_thresh`	float	**1.5**, $0 < x \leq 10^{10}$	for sky estimation, detection threshold above sky in units of σ above bkg for objects. The threshold is applied to the flux within an aperture of 10 pixel radius.
`psm_nbsize`	int	**256**, $1 \leq x \leq 2048$	for sky estimation, size of smoothing box in pixels when estimating bkg for objects
`psm_smkern`	float	**2.0**, $0 < x \leq 5$	for sky estimation, size of smoothing kernel in pixels
`cacheloc`	string	**.**, any string	directory where cache of calibration catalogue data ('.catcache') is stored relative to current working directory (.)
`magerrcut`	float	**100.0**, any float	objects in the photometric reference catalogue with a magnitude error greater than this value will be excluded from the calibration

8.5.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
BASIC_CALIBRATED_STD	exp_[N].fits	<FILE>_ex.fits	Calibrated image of each input standard star field
BASIC_VAR_MAP	exp_var_[N].fits	<FILE>_ex_var.fits	Variance map of BASIC_CALIBRATED_STD
BASIC_CAT_STD	exp_cat[N].fits	<FILE>_ex_cat.fits	Catalogue of objects in BASIC_CALIBRATED_STD
MEAN_SKY	sky_[N].fits	sky_<FIRSTFILE>.fits	Mean sky subtracted from STD images
MEAN_SKY_VAR	sky_var_[N].fits	sky_var_<FIRSTFILE>.fits	Variance map of MEAN_SKY
MATCHSTD_ASTROM	mstd_a[N].fits	<FILE>_ex_mstd_a.fits	Matched astrometric standard catalogue for BASIC_CALIBRATED_STD
MATCHSTD_PHOTOM	mstd_p[N].fits	<FILE>_ex_mstd_p.fits	Matched photometric standard catalogue for BASIC_CALIBRATED_STD

The index number [N] reflects the chronological order of the relevant files in the SoF, e.g. the file with the earliest MJD-OBS in the group has [N] = 1.

Note that <FIRSTFILE> is the first file listed in the SoF among the collection of files used to create that product.

8.5.5 Quality control parameters

The following QC parameters are found in each extension of the `BASIC_CALIBRATED_STD` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

`QC.AXT0`: [mag] Ext corrected zeropoint

`QC.ZPOINT`: [mag] Measured zeropoint without extinction

`QC.ZPOINT FWHM_AS`: [arcsec] Median FWHM of stars

The following QC parameters are found in each extension of the `BASIC_CAT_STD` files:

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in each extension of the `BASIC_VAR_MAP` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

The following QC parameters are found in each extension of the `MATCHSTD_ASTROM` files:

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

The following QC parameters are found in each extension of the `MATCHSTD_PHOTOM` files:

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The mean and RMS value of several of these keywords are stored in the PHU of the `MASTER_DARK` product. The average of the keyword value `QC.KW` across all detectors/extensions is stored as `QC.KW.AVG`; the RMS value is stored as `QC.KW.RMS`. The following AVG/RMS values are computed and stored in the PHU:

`QC.ZPOINT.AVG/RMS`

`QC.AXT0.AVG/RMS`

`QC.ZPOINT.TEL.AVG/RMS`

`QC.AXT0.TEL.AVG/RMS`

`QC.ZPOINT.FWHM_AS.AVG/RMS`

`QC.MAGZPT.AVG/RMS`

`QC.MAGZERR.AVG/RMS`

`QC.LIMITING.MAG.AVG/RMS`

`QC.SKYBRIGHT.AVG/RMS`

`QC.MAGNZPT.AVG/RMS`

`QC.ELLIPTICITY.AVG/RMS`

8.6 hawki_science_process

8.6.1 Description

Process a complete stack of HAWK-I data. Remove instrumental signature, remove sky background, combine jitters, photometrically and astrometrically calibrate the stacked image. A description of some of the common algorithms used by this recipe can be found in §9.

8.6.2 Input frames

Frame tag	Constraint	# files
OBJECT	Required	≥ 1
SKY	Optional	≥ 1
MASTER_DARK	Required	1
MASTER_TWILIGHT_FLAT	Required	1
MASTER_CONF	Required	1
MASTER_READGAIN	Required	1
PHOTCAL_TAB	Required	1
SCHLEGEL_MAP_NORTH	Required	1
SCHLEGEL_MAP_SOUTH	Required	1
MASTER_2MASS_CATALOGUE_ASTROM	Optional	1
MASTER_PPMXL_CATALOGUE_ASTROM	Optional	1
MASTER_LOCAL_CATALOGUE_ASTROM	Optional	1
MASTER_2MASS_CATALOGUE_PHOTOM	Optional	1
MASTER_PPMXL_CATALOGUE_PHOTOM	Optional	1
MASTER_LOCAL_CATALOGUE_PHOTOM	Optional	1
MASTER_OBJMASK	Optional	1
MATCHSTD_PHOTOM	Optional	1
MASTER_SKY	Optional	1
MASTER_SKY_VAR	Optional	1

Note that if only one OBJECT file is provided, a sky estimation is impossible. If two OBJECT files are provided, a sky can be created in principle but it is not likely to be accurate.

8.6.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
savemstd	bool	**false**, true	If true, create matched standard catalogues
savecat	bool	**false**, true	If true, create catalogues of objects detected in each input OBJECT image
skyalgo	string	**auto**, master, none, pawsky_mask, pawsky_mask_pre, pseudo_tilesky, simplesky_mask, pawsky_minus	See §9.7

continued on next page

Parameter	Type	Values **default**,other	Description
`preview_only`	bool	**false**, true	If true, don't run recipe but instead print out a summary of how data will be processed
`minphotom`	int	**1**, $1 \leq n \leq 10^5$	During photometric calibration, at least `minphotom` number of standards must be used.
`prettynames`	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.
`savemstd`	bool	**false**, true	If true, create matched standard catalogues
`cdssearch_astrom`	string	**none**, 2mass, usnob, ppmxl, wise	If not `none`, retrieve subset of specified catalogue from CDS for astrometric calibration
`cdssearch_photom`	string	**none**, 2mass, ppmxl	If not `none`, retrieve subset of specified catalogue from CDS for photometric calibration
`stk_cat_ipix`	int	**10**, $1 \leq n \leq 10^5$	min. number of pixels for catalogued objects in stack
`stk_cat_thresh`	float	**1.5**, $0 < x \leq 10^{10}$	detection threshold above sky in units of σ above bkg for catalogued objects in stack. The threshold is applied to the flux within each aperture.
`stk_cat_icrowd`	bool	**true**, false	if true, use deblending for catalogued objects in stack
`stk_cat_rcore`	float	**10.0**, $0 < x \leq 1024$	size of standard aperture in units of pixels for catalogued objects in stack
`stk_cat_nbsize`	int	**128**, $1 \leq x \leq 10^5$	size of smoothing box when estimating bkg for catalogued objects in stack
`stk_lthr`	float	**5.0**, $0 < x \leq 10^{10}$	when stacking, reject pixel more than `stk_lthr` units of σ below sky bkg

continued on next page

Parameter	Type	Values **default**,other	Description
`stk_hthr`	float	**5.0**, $0 < x \leq 10^{10}$	when stacking, reject pixel more than `stk_hthr` units of σ above sky bkg
`stk_method`	string	**linear**, nearest	when stacking, an RA/DEC is calculated for each input pixel. If `nearest`, the pixel in the output stack whose centre is nearest to that RA/DEC is given that input pixel. If `linear`, the exact pixel location of the RA/DEC on the output grid is calculated and the contribution from the input pixel is spread to the neighbouring 4 pixels that surround that point and is weighted by the fractional overlap.
`stk_seeing`	bool	**false**, true	when stacking, weight each image by seeing value
`stk_fast`	string	**auto**, fast, slow	See §9.1
`stk_nfst`	int	**30**, any int	See §9.1
`psm_ipix`	int	**10**, $1 \leq n \leq 10^5$	for pawsky_mask sky estimation, min. number of pixels for objects
`psm_niter`	int	**5**, $1 \leq n \leq 10$	for pawsky_mask sky estimation, number of iterations
`psm_thresh`	float	**1.5**, $0 < x \leq 10^{10}$	for pawsky_mask sky estimation, detection threshold above sky in units of σ above bkg for objects. The threshold is applied to the flux within an aperture of 10 pixel radius.
`psm_nbsize`	int	**128**, $1 \leq x \leq 2048$	for pawsky_mask sky estimation, size of smoothing box in pixels when estimating bkg for objects
`psm_smkern`	float	**2.0**, $0 < x \leq 5$	for pawsky_mask sky estimation, size of smoothing kernel in pixels

continued on next page

Parameter	Type	Values **default**,other	Description
`cacheloc`	string	**.**, any string	directory where cache of calibration catalogue data ('.catcache') is stored relative to current working directory (.)
`magerrcut`	float	**100.0**, any float	objects in the photometric reference catalogue with a magnitude error greater than this value will be excluded from the calibration

8.6.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
BASIC_CALIBRATED_SCI	exp_[N].fits	<FILE>_ex.fits	Calibrated image of each input OBJECT file
BASIC_VAR_MAP	exp_var_[N].fits	<FILE>_ex_var.fits	Variance map of each BASIC_CALIBRATED_SCI
BASIC_CALIBRATED_SKY	exp_[N].fits	<FILE>_ex.fits	Calibrated image of each input SKY file
BASIC_VAR_MAP_SKY	exp_var_[N].fits	<FILE>_ex_var.fits	Variance map of each BASIC_CALIBRATED_SKY
OBJECT_CATALOGUE_SCI	exp_cat[N].fits	<FILE>_ex_cat.fits	Catalogue of objects in each BASIC_CALIBRATED_SCI
MATCHSTD_ASTROM	exp_mstd_a[N].fits	<FILE>_ex_mstd_a.fits	Matched astrometric standard catalogue for each BASIC_CALIBRATED_SCI
MEAN_SKY	sky_[N].fits	sky_<FIRSTFILE>.fits	Mean sky subtracted from OBJECT images
MEAN_SKY_VAR	sky_var_[N].fits	sky_var_<FIRSTFILE>.fits	Variance map of MEAN_SKY
JITTERED_IMAGE_SCI	stack_1.fits	<FIRSTFILE>_st.fits	Calibrated stack of all OBJECT files
CONFIDENCE_MAP_JITTERED	stack_conf_1.fits	<FIRSTFILE>_conf.fits	Confidence map of JITTERED_IMAGE_SCI

continued on next page

PRO.CATG	Filename		Description
	Default	Pretty	
JITTERED_VAR_IMAGE	stack_var1.fits	<FIRSTFILE>_var.fits	Variance map of JITTERED_IMAGE_SCI
OBJECT_CATALOGUE_JITTERED	stack_cat_1.fits	<FIRSTFILE>_cat.fits	Catalogue of objects in JITTERED_IMAGE_SCI
MATCHSTD_ASTROM	stack_mstd_a1.fits	<FIRSTFILE>_st_mstd_a.fits	Matched astrometric standard catalogue for JITTERED_IMAGE_SCI
MATCHSTD_PHOTOM	stack_mstd_p1.fits	<FIRSTFILE>_st_mstd_p.fits	Matched photometric standard catalogue for JITTERED_IMAGE_SCI

The index number [N] reflects the chronological order of the relevant files in the SoF, e.g. the file with the earliest MJD-OBS in the group has [N] = 1.

Note that <FIRSTFILE> is generally the first file listed in the SoF among the collection of files used to create that product. However, if offset sky frames are provided, the <FIRSTFILE> for the OBJECT and SKY products may be different.

8.6.5 Quality control parameters

The following QC parameters are found in each extension of the `BASIC_CALIBRATED_SCI`, `CONFIDENCE_MAP_JITTERED`, and `JITTERED_VAR_IMAGE` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

The following QC parameters are found in each extension of the `BASIC_VAR_MAP` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

The following QC parameters are found in each extension of the `OBJECT_CATALOGUE_SCI` files:

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

The following QC parameters are found in each extension of the `JITTERED_IMAGE_SCI` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in each extension of the `OBJECT_CATALOGUE_JITTERED` and `MATCHSTD_PHOTOM` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in each extension of the `MATCHSTD_ASTROM` files:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

8.7 hawki_science_postprocess

8.7.1 Description

Process a complete stack of HAWK-I data and create a tile. Remove instrumental signature, remove sky background, combine jitters, photometrically and astrometrically calibrate the stacked image. A description of some of the common algorithms used by this recipe can be found in §9.

8.7.2 Input frames

Frame tag	**Constraint**	**# files**
BASIC_CALIBRATED_SCI	Required	≥ 1
BASIC_VAR_MAP	Required	Same as BASIC_CALIBRATED_SCI
MASTER_CONF	Required	1
PHOTCAL_TAB	Required	1
SCHLEGEL_MAP_NORTH	Required	1
SCHLEGEL_MAP_SOUTH	Required	1
MASTER_2MASS_CATALOGUE_ASTROM	Optional	1
MASTER_PPMXL_CATALOGUE_ASTROM	Optional	1
MASTER_LOCAL_CATALOGUE_ASTROM	Optional	1
MASTER_2MASS_CATALOGUE_PHOTOM	Optional	1
MASTER_PPMXL_CATALOGUE_PHOTOM	Optional	1
MASTER_LOCAL_CATALOGUE_PHOTOM	Optional	1
MATCHSTD_PHOTOM	Optional	1

8.7.3 Recipe parameters

Parameter	Type	Values **default**,other	Description
nebulise	bool	**false**, true	If true, object catalogue is derived from nebulised tile
minphotom	int	**20**, $1 \leq n \leq 10^5$	During photometric calibration, at least minphotom number of standards must be used.
prettynames	bool	**false**, true	If false, output files will have standard name. If true, files will have a more descriptive name.
cdssearch_astrom	string	**none**, 2mass, usnob, ppmxl, wise	If not none, retrieve subset of specified catalogue from CDS for astrometric calibration
cdssearch_photom	string	**none**, 2mass, ppmxl	If not none, retrieve subset of specified catalogue from CDS for photometric calibration
stk_fast	string	**auto**, fast, slow	See §9.1

continued on next page

Parameter	Type	Values **default**,other	Description
`stk_nfst`	int	**30**, any int	See §9.1
`savemstd`	bool	**false**, true	If true, create matched standard catalogues
`neb_medfilt`	int	**101**, $11 \leq n \leq 2047$	Median filter size in pixels for nebuliser
`neb_linfilt`	int	**33**, $3 \leq n \leq 2047$	Linear filter size in pixels for nebuliser
`cat_ipix`	int	**4**, $1 \leq n \leq 10^5$	min. number of pixels for catalogued objects
`cat_thresh`	float	**1.25**, $0 < x \leq 10^{10}$	detection threshold above sky in units of σ above bkg for catalogued objects. The threshold is applied to the flux within each aperture.
`cat_icrowd`	bool	**true**, false	if true, use deblending for catalogued objects
`cat_rcore`	float	**3.0**, $0 < x \leq 1024$	size of standard aperture in units of pixels for catalogued objects
`cat_nbsize`	int	**64**, $1 \leq x \leq 10^5$	size of smoothing box when estimating bkg for catalogued objects
`cacheloc`	string	**.**, any string	directory where cache of calibration catalogue data ('.catcache') is stored relative to current working directory (.)
`magerrcut`	float	**100.0**, any float	objects in the photometric reference catalogue with a magnitude error greater than this value will be excluded from the calibration

8.7.4 Product frames

PRO.CATG	Filename		Description
	Default	Pretty	
TILED_IMAGE	tile_1.fits	<FIRSTFILE>_ex_tl.fits	Calibrated tile of all BASIC_CALIBRATED_SCI files
TILED_OBJECT_CATALOGUE	tile_cat_1.fits	<FIRSTFILE>_ex_tl_cat.fits	Catalogue of objects in TILED_IMAGE
TILED_CONFIDENCE_MAP	tile_conf_1.fits	<FIRSTFILE>_ex_tl_conf.fits	Confidence map of TILED_IMAGE
MATCHSTD_ASTROM	tile_mstd_a1.fits	<FIRSTFILE>_ex_tl_mstd_a.fits	Matched astrometric standard catalogue for TILED_IMAGE
MATCHSTD_PHOTOM	tile_mstd_p1.fits	<FIRSTFILE>_ex_tl_mstd_p.fits	Matched photometric standard catalogue for TILED_IMAGE
TILED_VAR_MAP	tile_var1.fits	<FIRSTFILE>_ex_tl_var.fits	Variance map of TILED_IMAGE

Note that <FIRSTFILE> is the first file listed in the SoF among the collection of files used to create that product.

8.7.5 Quality control parameters

The following QC parameters are found in the primary header of the `TILED_IMAGE` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

`QC.PROC.SCHEME`: Character string denoting the phase-3 release; currently 'HAWKI_R'

`QC.RUN.INFO`: Hyperlink to the data run info

`QC.RUN.INFO`: Hyperlink to the archive file list

`QC.ABSTRACT`: Hyperlink to the data program abstract

`QC.PLOT`: Hyperlink to the QC plot

`QC.AIRM.MEAN`: Average of: `TEL.AIRM.START` of the first input `BASIC_CALIBRATED_SCI` file to hawki_science_postprocess; and, `TEL.AIRM.END` of the last input `BASIC_CALIBRATED_SCI` to hawki_science_postprocess

`QC.FILT1.NAME`: Value of `INS.FILT1.NAME` with trailing blanks removed

`QC.FILT2.NAME`: Value of `INS.FILT2.NAME` with trailing blanks removed

`QC.NCORRS.NAME`: Value of `DET.NCORRS.NAME` with trailing blanks removed

`QC.MAGZPT.CORR`: Computed as $\texttt{QC.MAGZPT} + 2.5 \times \log_{10}(\texttt{DET.NDIT})$

`QC.FLAG.TPL`: Set to 0 if the number of input `OBJECT` and `SKY` frames to hawki_science_process equals `TPL.NEXP`; otherwise set to 1

`QC.FLAG.SKY_YN`: Set to 'Y' if input to hawki_science_process contains a `SKY` exposure; otherwise set to 'N'

`QC.FLAG.AO_YN`: Set to 'Y' if `INS.MODE` contains the string 'AO_IMG'; otherwise set to 'N'

The following QC parameters are found in the first extension of the `TILED_OBJECT_CATALOGUE` and `MATCHSTD_PHOTOM` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

`QC.MAGZPT`: [mag] photometric zeropoint

`QC.MAGZERR`: [mag] photometric zeropoint error (1.48 * median absolute deviation of MAGZPT among all extensions). Set to 1.0 if photometric calibration failed.

`QC.LIMITING_MAG`: [mag] 5 sigma limiting mag.

`QC.SKYBRIGHT`: [mag/arcsec**2] sky brightness

`QC.MAGNZPT`: number of stars available for magzpt calc; that actual number used may be less than this, depending on the value of `magerrcut`

The following QC parameters are found in the primary header of the `TILED_CONFIDENCE_MAP` and `TILED_VAR_MAP` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

The following QC parameters are found in the first extension of the `MATCHSTD_ASTROM` file:

`QC.WCS_DCRVAL1`: [deg] change in crval1 wrt values in original header

`QC.WCS_DCRVAL2`: [deg] change in crval2 wrt values in original header

`QC.WCS_DTHETA`: [deg] change in rotation wrt values in original header

`QC.WCS_SCALE`: [arcsec] mean plate scale

`QC.WCS_SHEAR`: [deg] abs(xrot) - abs(yrot) where xrot & yrot are the rotation angles in each axis as calculated by the WCS rotation matrix

`QC.WCS_RMS`: [arcsec] Average error in WCS fit

`QC.SATURATION`: [adu] Saturation level

`QC.MEAN_SKY`: [adu] Median sky brightness

`QC.SKY_NOISE`: [adu] Pixel noise at sky level

`QC.IMAGE_SIZE`: [arcsec] Average FWHM of stellar object

`QC.ELLIPTICITY`: Average stellar ellipticity (1-b/a)

`QC.POSANG`: [degrees] Median position angle (from North)

`QC.APERTURE_CORR`: Stellar ap-corr 1x core flux

`QC.NOISE_OBJ`: Number of noise objects, where a noise object is one that the catalogue has classified as an object that may not be astrophysical

9 Recipe Algorithms

9.1 Stacking

The stacking module used by the HAWK-I pipeline recipes for the science observations has two modes of operation, a `slow` and a `fast` algorithm. The algorithm needs to stack not just the science data, but also the science variance data. It creates an output stack, output stack variance and an output stack confidence map. The volume of data that is read and written can make the memory required quite large and this can be increased further if the jitter offsets are large too.

The 'fast' algorithm can be used when stacking a small number of images. As the name implies it is usually pretty quick, but it is very greedy with memory. For stacking problems with more images it is better to use the slow algorithm. The recipes that use the stacking module also offer an *auto* mode. This allows the recipe to decide which algorithm to use. If the number of input frames is less than or equal to `stk_nfst`, the 'fast' mode is used.

When stacking, an RA/DEC is calculated for each input pixel. If `stk_method = nearest`, the pixel in the output stack whose centre is nearest to that RA/DEC is given that input pixel. If `stk_method = linear`, the exact pixel location of the RA/DEC on the output grid is calculated and the contribution from the input pixel is spread to the neighbouring 4 pixels that surround that point and is weighted by the fractional overlap.

Changes in throughput (i.e. magnitude zero point) and/or seeing between consecutive raw science frames within an OB are not taken into account as part of the stacking procedure. Users are advised to check for changes in seeing or photometric conditions in each image before stacking; no warning are issued by the recipe. Aberrant images may be discarded from the input list to avoid unexpected results.

9.2 Variance maps

All science recipes in the HAWK-I pipeline package propagate the uncertainty of the flux of each pixel in each of the science images. This is done using the standard equations for the propagation of uncertainties. The initial estimate of variance is based on the standard Poisson model, modified by the number of DITs and the number of samples in the non-destructive read. If a pixel is flagged as bad, then its variance estimate is defined as zero.

The contribution to the variance from the readnoise is calculated as:

$$\sigma_{var}^2 = 12 * \frac{\sigma_{ro}^2(\texttt{DET.NDSAMPLES} - 1)}{\texttt{DET.NDSAMPLES} * (\texttt{DET.NDSAMPLES} + 1) * \texttt{DET.NDIT}} \tag{1}$$

where σ_{ro} is the readnoise (in units of electrons) defined in §9.5.

An extra factor that accounts for the modification to the standard Poisson noise model is:

$$f = 1.2 * \frac{\texttt{DET.NDSAMPLES}^2 + 1}{g * \texttt{DET.NDSAMPLES} * (\texttt{DET.NDSAMPLES} + 1) * \texttt{DET.NDIT}} \tag{2}$$

where g is the gain (in units of electrons per ADU) defined in §9.5.

The total variance for each pixel in an image is then calculated as

$$\sigma^2(x, y) = \sigma_{var}^2 + f * |I(x, y)| \tag{3}$$

where $I(x, y)$ is the pixel value in the image.

9.3 Confidence maps

A confidence map is similar to a weight map. These are initially derived from the master twilight flat field for the detector and hence gives an indication of the relative quantum efficiency of each pixel, rather than an indication of the poisson uncertainty in the pixel flux. The map is stored as an integer array that has been normalised to a mean of 100. Bad pixels (either dead or hot) are given a confidence value of zero. If images are stacked, then the confidence map is propagated and can be used as a weight map or an exposure time map.

9.4 WCS fitting

Cartesian and equatorial coordinates are fitted to standard plate solution models with 6 constants that account for non-zero shear and scale differences. The difference in the predicted x,y coordinates and the true x,y coordinates is used to adjust the tangent point first to correct for telescope pointing error (i.e. `RA` and `DEC` keywords in raw file PHU) . The median difference of the equatorial coordinates between that implied from the two sets of Cartesian coordinates is used to update the tangent point. A full least-squares solution is performed and the results are written back to the given FITS WCS header structure (using a TAN projection).

For a 6 constant model, fits are done with the input standards for the equations:

$$\xi = ax + by + c \tag{4}$$

$$\eta = dx + ey + f \tag{5}$$

where ξ, η are standard coordinates with respect to the tangent point (transformed onto the detector system), to find values of a, b, c, e, d and f.

9.5 Detector readnoise and gain

The read noise and gain is measured using two twilight flat frames of similar illumination and two dark frames (all with identical DIT and NDIT values). First, a robust estimate of the variance in an image of the difference between the two flat frames is calculated (σ_f^2). The same is done for the dark frames (σ_d^2). A correction for any inter-pixel correlation is made by evaluating the scalar auto-correlation among pixels in the flat frames (ξ). If the mean background of the flat and dark frames are m_{f1}, m_{f2}, m_{d1}, and m_{d2}, the gain (in electrons per ADU) is:

$$g = \frac{1}{\xi} \frac{(m_{f1} + m_{f2}) - (m_{d1} + m_{d2})}{\sigma_f^2 - \sigma_d^2} \tag{6}$$

and the readout noise (in electrons) is:

$$\sigma_{ro} = \frac{g\sigma_d}{\sqrt{2}} \tag{7}$$

9.6 Object detection/catalogue generation

In order to provide quality control, and astrometric and photometric calibration information, the recipes generate detected object (i.e. stars, galaxies) catalogues for each target frame.

Objects are detected and parameterised using the processed images and confidence maps. A high-level summary of this process is:

- estimate the local sky background over the field and track any variations at adequate resolution to eventually remove them
- detect objects/blends of objects and keep a list of pixels belonging to each blend for further analysis (see [6] for details).
- parameterise the detected objects, i.e. perform astrometry, photometry and some sort of shape analysis.

Note that pixels are weighted in the catalogue generation according to their value in the confidence map. Hence if a pixel is marked as bad, then it is not included in the aperture flux. The number of bad pixels with an aperture is reported in the 'Error_bit_flag' column of the output table. The presence of bad pixels will also be reflected in the average confidence for the aperture (column 'Av_conf').

9.6.1 Background analysis and object detection

The possibly-varying sky background is estimated automatically, prior to object detection, using a combination of robust iteratively-clipped estimators.

Any variation in sky level over the frame is dealt with by forming a coarsely sampled background map grid. Within each background grid pixel (specified by recipe parameter `stk_cat_nbsize`), an iteratively k-sigma clipped median value of 'sky' is computed based on the histogram of flux values within the grid pixel zone. A robust estimate of sigma can be computed using the Median of the Absolute Deviation (MAD) from the median. This will then be further processed to form the frame background map.

After removing the (possibly) varying background component, a similar robust estimate of the average sky level and sky noise per pixel can be made. This forms part of the quality control measures and also helps to robustly determine the detection threshold for object analysis.

Individual objects are detected using a standard matched filter approach. Since the only images difficult to locate are those marginally above the sky noise, assuming constant noise is a good approximation (after factoring in the confidence map information) and the majority of these objects will have a shape dominated by the point spread function (PSF), which thereby defines the filter to use.

9.6.2 Object parameterisation

The following image parameters are computed efficiently for each detected object:

Isophotal intensity: integrated flux within the boundary defined by the threshold level, i.e. the 0th object moment

$$I_{iso} = \sum_i I(x_i, y_i) \tag{8}$$

For objects where $I(x_i, y_i)$ is well described by a Gaussian, the isophotal intensity is related to the total intensity by the factor $(1 - I_t/I_p)^{-1}$, where I_p is the peak flux and I_t is the threshold level (all relative to sky).

Position: intensity-weighted location of object on the image, i.e. 1st moment:

$$x_0 = \sum_i xI(x_i, y_i)/I_{iso} \tag{9}$$

$$y_0 = \sum_i yI(x_i, y_i)/I_{iso} \tag{10}$$

Covariance matrix: triad of intensity-weighted 2nd moments used to estimate the eccentricity/ellipticity, position angle, and intensity-weighted size of an object:

$$\sigma_{xx} = \sum_i (x_i - x_0)^2 I(x_i, y_i)/I_{iso} \tag{11}$$

$$\sigma_{xy} = \sum_i (x_i - x_0)(y_i - y_0) I(x_i, y_i)/I_{iso} \tag{12}$$

$$\sigma_{yy} = \sum_i (y_i - y_0)^2 I(x_i, y_i)/I_{iso} \tag{13}$$

The shape parameters are derived by considering these quantities in relation to an elliptical Gaussian function with the same 2nd moments. The scale size ($\sqrt{\sigma_{rr}}$) is computed with $\sigma_{rr} = \sigma_{xx} + \sigma yy$. The eccentricity is $f = \sqrt{(\sigma_{xx} - \sigma_{yy})^2 + 4\sigma_{xy}}/\sigma_{rr}$. The position angle θ is computed with $tan(2\theta) = 2\sigma_{xy}/(\sigma_{yy} - \sigma_{xx})$. The ellipticity ($e$) is simpler to interpret for estimation potential image distortions (e.g. trailing) and is related to the eccentricity by $e = 1 - \sqrt{(1-f)/(1+f)}$.

Areal profile: a variant on the radial profile, which measures the area of an object at various intensity levels. Unlike a radial profile, which needs a prior estimate of the object centre, the areal profile provides a single-pass estimate of the object area at m discrete intensity levels $T + p_j$ where $p_j; j = (1...m)$ are intensity levels relative to threshold T (usually spaced logarithmically to give even sampling). The threshold T can be specified using the recipe parameter `stk_cat_thresh`.

Peak height: a useful related addition to the areal profile information and is defined as $I_p = max(I(x_i, y_i))$ or alternatively measured by extrapolation from the areal profile if the image is saturated.

Seeing: The areal profile provides a direct method to estimate the seeing of objects in an image by enabling the average area of stellar images (point sources) at half the peak height ($< A >$) to be estimated. The seeing, or FWHM, is then given by $FWHM = 2\sqrt{< A > /\pi}$.

Aperture flux: a series of aperture fluxes are required for object morphological classification. Aperture flux is defined as the integrated flux with some radius r of the object centre:

$$I_{ap} = \sum_{i \in r}^{N} I_i - N \times sky \tag{14}$$

where boundary pixels are weighted pro-rate (soft-edged aperture photometry). A series of these is used to define the curve-of-growth (I_{ap} vs. r) for each object.

The error in the aperture flux is calculated as:

$$\sigma_{ap} = \sqrt{I_{ap}/gain + \pi r^2(\sigma^2_{globalsky} + \sigma^2_{localsky})} \tag{15}$$

where $\sigma_{localsky}$ is a robust estimate of the sky variance among adjacent cells (defined by `stk_cat_nbsize`), $\sigma_{globalsky}$ is a robust estimate of the sky variance over the whole image, and the $gain$ is the value calculated by the `detector_noise` recipe.

In order to estimate the total flux of an object, an aperture correction must be applied. This correction is calculated from the curve-of-growth analysis and appears in the FITS header of the catalogue (`APCOR_i` is the correction to be applied to the ith aperture fluxaperture correction for each pa

To convert aperture fluxes to magnitudes, an aperture correction must be applied. This correction is calculated from the curve-of-growth and accounts for missing flux in a particular aperture. For example, the magnitude of an object using the flux in the third bin ($i = 3$) may be calculated as

$$m = \texttt{MAGZPT} - \texttt{2.5}\log_{10}(\texttt{APER_FLUX_3}/\texttt{EXPTIME}) - \texttt{APCOR3}) \tag{16}$$

where `MAGZPT, EXPTIME,` and `APCOR3` appear in the FITS header of the catalogue file; `APER_FLUX_3` is a column in the FITS binary table of the catalogue file.

For overlapping objects that are deblended, the boundaries are in practice simultaneously fitted top-hat functions (to minimise the effects of crowding). Object external to the blend are also flagged and not included in the large radius summations.

9.6.3 Morphological classification

The recipes produce a series of background-corrected flux measures for each object in a set of 'soft-edged' apertures of radius $r/2, r/\sqrt{2}, r, r\sqrt{2}, 2r, ...12r$, where r is specified by the recipe parameter `stk_cat_rcore`. Generally it is recommended that `stk_cat_rcore` be fixed as the median seeing for the site + telescope + camera. For HAWK-I, the default value of `stk_cat_rcore` is 10.0 pixels (1.06″). The average curve-of-growth for stellar objects is used to define automatically an aperture correction for each aperture used and also forms the basis for object morphological classification (required for isolating stellar images for seeing and trailing quality control).

The curve-of-growth of the flux for each object is compared with that derived from the (self-defining) locus of stellar objects, and combined with information on the ellipticity of each object, to generate the classification statistic. This statistic is designed to preserve information on the 'sharpness' of the object profile and is re-normalised, as a function of magnitude, to produce the equivalent of an $N(0, 1)$ measure, i.e. a normalised Gaussian of zero-mean and unit variance. Objects lying within 2-3σ are generally flagged as stellar images, those below $\approx 2\sigma$ (i.e. sharper) as noise-like, and those above 2-3σ (i.e. more diffuse) as non-stellar. A by-product of the curve-of-growth analysis is the estimate of the average PSF aperture correction for each detector.

9.7 Sky Estimation

Estimating the sky background well is one of the hardest jobs in infrared imaging. The infrared sky varies both spatially and temporally over short time scales. The jittering of exposures is a method used to allow for the observational data to be self-calibrating with respect to the estimation of the background emission. The basic idea is the stack all object images without dithering but with rejection in order to remove all astronomical objects from the background stack. However if astronomical objects are included in the observation that are larger than the typical jitter step, then it is possible that the rejection will fail and the background stack will be contaminated by emission from such objects.

For this reason the HAWK-I science recipe contains a number background estimation algorithms. Any of them can be chosen using the `skyalgo` command line switch. There is also a mode `skyalgo=auto` where the recipe will choose what it believes to be the best algorithm. Users can see which algorithm is chosen by running the recipe with `preview_only = true`). The automatically selected algorithm is also reported in the `DRS.SKYALGO` keyword in each extension of the `MEAN_SKY` recipe product.

In the situation with large extended objects in the science observations, there is also the option to include offset sky exposures in the SoF. These are exposures that are well away from the object of interest and within the same OB as the science exposures. Below is a description of each of the algorithms supported by the `hawki_science_process` recipe.

Note that when using any algorithm (other than `none`), the recipes may create more than one sky frame from a set of input filse. If the elapsed time between the first and last frame is more than $\approx$ 30 minutes, the set is split apart into chunks. A sky is created from each 'chunk' only if there are more than $\approx$ 8 members of each chunk. If not, some chunks are merged together to ensure a reasonable sky is created.

`pawsky_mask`: The OB is of a simple jittered stack. The algorithm starts by doing a simple stack with rejection of all the science images to form an initial background image. The exposures are then all background corrected using this image and then combined using the jitter offsets to form a stacked image. An object detection is performed on this stacked image to create an object mask. The stacked image, initial sky background map and the sky corrected observations are thrown away. Now a new background map is formed by doing a stack of the input images with rejection, but this time also using the object mask we've just created. The procedure re-estimation of the background and masking object pixels continues until the number of masked pixels converges.

`pawsky_mask_pre`: This is the same as the pawsky_mask algorithm, except that a master object mask is given in the SoF. In this case there is no need to iterate. It is absolutely crucial for the object mask to (1) cover the whole area of sky covered by the observations and (2) to have an accurate world coordinate system represented in its FITS header.

`pawsky_minus`: Very faint objects below the detection threshold for an observation may not reject out in a simple "stack with rejection" estimation algorithm. The pawsky_minus algorithm seeks to fix this by using a master object mask as with pawsky_mask_pre during the background estimation. But the difference is that for each observation a separate sky background image is formed where the observation in question has been removed from the stack. Because this algorithm creates a separate background estimate for each science frame and to avoid creating a clutter sky images, the algorithm only outputs corrected frames and a single representative sky background image.

`master`: If this option is chosen then a master background map should be included in the SoF. This must be used with extreme caution as a master sky is unlikely to be a good match for a given OB unless they are spatially and temporally close to each other.

`pseudo_tilesky`: There have been in the past instances of offset skies being taken not just in one location away from the science stack, but rather in several locations. These tend to manifest themselves as a separate template for each location in a given OB. In this instance the pseudo_tilesky provides a fast method for estimating the sky. This is done by creating intermediate background estimates by combining sets of observations containing one frame from each offset template. Then the intermediate background frames are combined into a final background frame. NB: this method is only to be used when offset sky images are present in the SoF and that these are jittered as described above in several different locations around the science image.

`simplesky_mask`: This method is used when the jitter offsets in the science images are large. The pawsky_mask algorithm would become slow (and somewhat suspect) in this situation as the dithering would create very large maps. The simplesky_mask method starts in the same way as the pawsky_mask algorthm, with a simple stack to create an initial background estimate and a correction of the science images using this initial estimate. An source detection is then done on each corrected science image to create a separate object pixel mask for each science image. These are used to mask out objects in the combination to create a new background map in the next iteration. The procedure iterates until the number of object pixels converges.

`none`: If this is chosen, then no background correction will be done at all.

`auto`: If this is chosen then the recipe attempts to make a decision on which is the best algorithm based on what is in the SoF.

9.8 Nebulising

For observations with a large amount of nebulosity it can be very difficult to get a good estimate for the sky background. This is where the *nebuliser* can be useful. The nebuliser calculates sliding means and medians of an image to create a background image, which is essentially an unsharp mask. This image is subtracted from the original to produce an image with greatly reduced nebulosity.

If the user chooses to use the nebuliser in one of the science recipes, then the stacked image is corrected for nebulosity. The nebulised image is used to create the source catalogue and then thrown away. The output images will be the un-nebulised stack with its accompanying confidence and variance map along with the source catalogue derived from the nebulised image.

A File Formats

A.1 Static Calibration Files

The `REFERENCE` darks and flats are pipeline products of `hawki_dark_combine` and `hawki_twilight_flat_combine`, respectively. After creation, these files have the keyword `PRO.CATG` inserted into the PHU of each file using the supplied `update_procatg.py` script. The values of these keywords are `REFERENCE_DARK` and `REFERENCE_TWILIGHT_FLAT`, respectively.

The `MASTER_READGAIN` file `readgain.fits` is the product of `hawki_detector_noise`.

The dust map files (`SFD_dust_4096_ngp.fits` and `SFD_dust_4096_ngp.fits`) are similar to the original FITS files provided by [7]. However, these files have the keyword `PRO.CATG` inserted into the PHU of each file using the supplied `update_procatg.py` script. The values of these keywords are `SCHLEGEL_MAP_NORTH` and `SCHLEGEL_MAP_SOUTH`, respectively.

The photometric calibration table file (`hawki_phot.fits`) is used to transform the instrumental magnitudes to a calibrated system. It is a binary FITS table with one extension per system. If using a local copy of a catalogue (e.g. 2MASS), an extension **must** be present that matches the name of the extension in the photometric catalogue index file (e.g. 'casu_2mass'). If using CDS, the name of one extension **must** match the value of the `cdssearch_photom` recipe parameter (e.g. '2mass').

A description of the columns for each extension is:

Column	Name	Type	Description
1	filter_name	char	Name of the filter. It must match the value of the `INS.FILT?.NAME` keyword from the data being calibrated.
2	atm_extcoef	float (1E)	The extinction coefficient for airmass of unity for the given filter.
3	mag_offset	float (1E)	A scalar value to be added to the instrumental magnitude once the colour equation has been applied.
4	coleq_columns	char	The name(s) of magnitude columns from the photometric standards catalogue to be used in the colour equations, e.g. 'Jmag'. The names must be separated by a comma.
5	coleq_errcols	char	The name(s) of magnitude error columns from the photometric standards catalogue, e.g. 'e_Jmag'. There must be one name for each of the column names given in the coleq_columns field. Any objects with an error in any of the coleq_errcols greater than the value specified by the recipe parameter `magerrcut` will be excluded from the calibration. The names must be separated by a comma.
6	coleq_coefs	char	The colour equation coefficients. There must be one coefficient for each of the column names given in the coleq_columns field. This should be formatted as a sequence of floating point numbers separated by commas.
7	gal_extcoef	float (1E)	Multiply the E(B-V) estimate from the Schlegel maps by `gal_extcoef` to give the total galactic absorption in `filter_name`
8	default_zp	float (1E)	If the photometric calibration fails, this default zero point value is used.
9	default_zp_err	float (1E)	If photometric calibration fails, this default error on the zero point value is used.

continued on next page

Column	Name	Type	Description

A.2 Astrometric and Photometric Catalogue Files

A user may choose to calibrate their data using data on disk in the form of index files and catalogue data files. An index file is a map between celestial coordinates and the filename where the standard object data resides. It is provided in a SoF to tell a recipe where to find the standards to use in the calibration.

The index file is a binary FITS table where the extension name describes the catalogue, e.g. 'casu_2mass'. It is necessary to set the `RFXNAME` in the primary header to `casu_local_astrom` or `casu_local_photom` for ESO Reflex to recognize the catalog. The columns are:

Column	Name	Type	Description
1	`filename`	char	Name of the file where data reside over a particular coordinate range. The file (or a link to it) must be found in the same directory as the index file.
2	`ramin`	float (1E)	Objects in `filename` have $\alpha >$ `ramin`.
3	`ramax`	float (1E)	Objects in `filename` have $\alpha \leq$ `ramax`
4	`decmin`	float (1E)	Objects in `filename` have $\delta >$ `decmin`
5	`decmax`	float (1E)	Objects in `filename` have $\delta \leq$ `decmax`

Note that all floating point numbers **must** be represented with 4-bytes (1E).

The catalogue data files are also binary FITS files. It must contain the columns described below; other columns are allowed but ignored. Note that the column names must appear as described below:

Column	Name	Type	Description
1	RA	float (1D)	Right ascension of the object (J2000). This **must** be represented as an 8-byte floating point number.
2	Dec	float (1D)	Declination of the object (J2000). This **must** be represented as an 8-byte floating point number.
3	<mag>	float (1E)	Magnitude of object in a particular filter. The name of this column must match one of those in the coleq_columns column in the photometric calibration table, e.g. 'Jmag'. There must be one <mag> column for each unique value of coleq_columns. The data in this column **must** be represented as a 4-byte floating point number. NULL values are allowed.
4	e_<mag>	float (1E)	Error in magnitude of object in a particular filter. The value of <mag> must match one of those in the coleq_columns column in the photometric calibration table, e.g. 'e_Jmag'. There must be one e_<mag> column for each unique value of coleq_columns. The data in this column **must** be represented as a 4-byte floating point number. NULL values are allowed.

A.3 Recipe Product Catalogues

The derived object catalogues are stored in multi-extension FITS files as binary tables, one for each image extension. Each detected object has an attached set of descriptors, forming the columns of the binary table, and summarising derived position, shape and intensity information (see section 9.6 for more details).

The following columns are present:

Column	Name	Description
1	Sequence_number	Running number for ease of reference, in strict order of image detections
2	Isophotal_flux	Standard definition of summed flux within detection isophote, apart from detection filter which is used to define pixel connectivity and hence which pixels to include. This helps to reduce edge effects for all isophotally derived parameters.
3 4 5 6	X_coordinate X_coordinate_err Y_coordinate Y_coordinate_err	The x, y intensity-weighted isophotal centre of gravity coordinates and errors with (1, 1) defined to be the centre of the first active pixel in the image array.
7 8 9	Gaussian_sigma Ellipticity Position_angle	Derived from second moment parameters. Equivalence between parameters and generalised elliptical Gaussian distribution is used to derive sigma = $(\sigma_a^2 + \sigma_b^2)^{1/2}$, ellipticity = $1.0 - \sigma_a/\sigma_b$, and position angle = angle of ellipse major axis wrt x axis in degrees
10 11 12 13 14 15 16 17	Areal_1_profile Areal_2_profile Areal_3_profile Areal_4_profile Areal_5_profile Areal_6_profile Areal_7_profile Areal_8_profile	The number of pixels above a series of threshold levels, relative to local sky. The levels are set at T, 2T, 4T, 8T, 16T, 32T, 64T and 128T where T is the analysis threshold (`stk_cat_thresh`). These can be thought of as a sort of poor man's radial profile. Note that for deblended, i.e. overlapping objects, only the first areal profile is computed and the rest are set to -1 (flagging the difficulty of computing accurate profiles). For blended images, Areal profile 8 is used to flag the start of the sequence of the deblended components by setting the first in the sequence to 0.
18 19	Peak_height Peak_height_err	Peak intensity and its error in ADU relative to local value of sky. Equivalent to zeroth-order aperture flux.
20 21 22 23 24 25 26 27	Aper_flux_1 Aper_flux_1_err Aper_flux_2 Aper_flux_2_err Aper_flux_3 Aper_flux_3_err Aper_flux_4 Aper_flux_4_err	Flux and error within a specified radial aperture where r_{core} =`stk_cat_rcore` and different rows correspond to (0.5, 1/$\sqrt{2}$, 1, $\sqrt{2}$, 2, 2$\sqrt{2}$, 4, 5, 6, 7, 8, 10, and 12) times r_{core}. These are a series of different radii soft-edged apertures designed to adequately sample the curve-of-growth of the majority of objects and to provide fixed-sized aperture fluxes for all objects. For example, in $\approx 0.8''$seeing, an r_{core} = $1''$aperture contains roughly 75% of the total flux of stellar images. The aperture fluxes are sky-corrected integrals (summations) with a soft-edge (i.e. pro-rata flux division for boundary pixels). However, for overlapping objects they are more subtle than this since they are in practice simultaneously fitted top-hat functions, to minimise the effects of crowding. Objects external to the blend are also flagged and not included in the large radius

continued on next page

Column	Name	Description
28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45	Aper_flux_5 Aper_flux_5_err Aper_flux_6 Aper_flux_6_err Aper_flux_7 Aper_flux_7_err Aper_flux_8 Aper_flux_8_err Aper_flux_9 Aper_flux_9_err Aper_flux_10 Aper_flux_10_err Aper_flux_11 Aper_flux_11_err Aper_flux_12 Aper_flux_12_err Aper_flux_13 Aper_flux_13_err	
46	Petr_radius	Petrosian radius r_p in pixels as defined in Yasuda, et al. 2001, AJ, 112, 1104.
47	Kron_radius	Kron radius r_k in pixels as defined by Bertin and Arnouts 1996, A&A Supp, 117, 393.
48	Half_radius	r_h estimate of object half-light radius.
49 50	Petr_flux Petr_flux_err	Petrosian flux and error to $2r_p$.
51 52	Kron_flux Kron_flux_err	Kron flux and error to $2r_k$.
53 54	Half_flux Half_flux_err	flux and error within half-light radius.
55	Error_bit_flag	Number of bad pixels within aperture of radius r_{core} - note this can be fractional due to soft-edged apertures
56	Sky_level	Local interpolated sky level from background tracker
57	Sky_rms	Local estimate of variation in sky level around object

continued on next page

Column	Name	Description
58	Av_conf	Average confidence level within r_{core} aperture. Useful for spotting spurious outliers in various parameter selection spaces
59 60	RA Dec	Single-precision (4 byte) RA and Dec of each object in degrees (only accurate to 50mas!). Astrometry should be derived more precisely from WCS in header and XY in columns 5 & 6.
61	Classification	Simple flag indicating most probable classification for object: [-2: Object is compact (maybe stellar), -1: Object is stellar, 0: Object is noise, 1: Object is non-stellar]. Saturated objects can be flagged by comparing the peak height + local sky with the `SATURATE` keyword in the header.
62	Statistic	An equivalent N(0,1) measure of how stellar-like an image is. It is used in deriving the classification (column 61) in a 'necessary but not sufficient' sense. This statistic is computed from a discrete curve-of-growth analysis from the peak and aperture fluxes and also factors in ellipticity information. The stellar locus is used to define the 'mean' and 'sigma' as a function of magnitude such that the 'statistic' can be normalised to an approximate N(0,1) distribution. See Irwin et al 1994 (SPIE 5493 411) for more details.
63–80	blank	